IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

INOVIS USA, INC.,	)	
	)	
Plaintiff,	)	
	)	
v.	)	C.A. No. 07-459-GMS
	)	
CLASSIFIED INFORMATION, INC. and DISTANCE DIGITAL LLC,	)	
	)	
	)	
Defendants.	)	

PLAINTIFF INOVIS USA, INC.'S OPENING BRIEF IN SUPPORT OF ITS MOTION TO STAY ALL PROCEEDINGS PENDING DETERMINATION OF ITS FILED REQUEST FOR REEXAMINATION

MORRIS, NICHOLS, ARSHT & TUNNELL LLP
Jack B. Blumenfeld, Esquire (#1014)
Julia Heaney, Esquire (#3052)
1201 North Market Street
P.O. Box 1347
Wilmington, DE 19899-1347
(302) 658-9200
jblumenfeld@mnat.com
jheaney@mnat.com
Attorneys for Plaintiff Inovis USA, Inc.

OF COUNSEL:

David J. Wolfsohn
Erich M. Falke
Jordan L. Jonas
WOODCOCK WASHBURN LLP
Cira Centre, 12th Floor
2929 Arch Street
Philadelphia, PA 19104
(215) 568-3100

Dated: April 28, 2008

TABLE OF CONTENTS

TABLE OF AUTHORITIES
FEDERAL CASES

FEDERAL STATUTES

NATURE AND STAGE OF THE PROCEEDINGS

On February 19, 2008, Classified filed a Counterclaim against Inovis for infringement of U.S. Patent No. 5,812,669 ("the 669 Patent"). (D.I. 48). On March 10, 2008, Inovis moved to dismiss Classified's Counterclaim for lack of subject matter jurisdiction. (D.I. 50). Classified filed its opposition brief on April 2 (D.I. 57), and Inovis' reply brief was filed on April 24, 2008 (D.I. 61). That motion is currently pending before this Court. On April 14, 2008—less than two months after Classified filed its Counterclaim for patent infringement—Inovis filed a Request for Reexamination with the United States Patent and Trademark Office ("PTO") of the 669 Patent based on prior art recently discovered by Inovis and not disclosed by the alleged inventors of the 669 Patent to the PTO. This motion seeks to stay the case in its entirety until the PTO determines whether it will grant the reexamination request and, if the PTO grants the request for reexamination, to stay the case until the PTO completes its reexamination.[1]

SUMMARY OF THE ARGUMENT

It is well-accepted that, where a defendant files for reexamination of the patent-in-suit, a stay of the proceedings in district court is appropriate. This is because a stay usually will result in substantial savings of time for the Court, and time and expense for the parties, as well as clarify issues if the patent survives reexamination. The stay requested here will provide multiple benefits. The PTO is highly likely to grant reexamination since it is granting reexamination in over 95% of applications. The PTO is also likely to invalidate the 669 Patent because the prior art submitted overwhelmingly supports that conclusion. If that occurs, the entire course of the

[1] If the PTO grants Inovis' request for reexamination, Inovis respectfully requests that the Court consider the PTO's actions and stay the entire case until the PTO completes its reexamination. *See Gioello Enters. Ltd. v. Mattel, Inc.*, No. 99-375 GMS, 2001 U.S. Dist. LEXIS 26158, at *1-2 & n.1 (D. Del. Jan. 29, 2001) (considering the PTO's grant of defendant's request for reexamination even though defendant filed its motion to stay before the PTO granted the request).

litigation of Classified's Counterclaim would be avoided, since the Counterclaim was just filed two months ago, and discovery has just started, with Classified having served its first set of discovery requests only last week. The Court would not have to hold a *Markman* hearing, construe the pertinent claim terms, consider cross motions for summary judgment on infringement and invalidity, hold a trial, or decide post-trial motions.

Even if the PTO only narrows one or more of the 669 Patent claims, the Court could take advantage of any changes to the patent claim scope that will most likely occur during the reexamination process. Any such changes in claim scope will affect the liability and damages issues for trial. Moreover, the Court will have the benefit of the PTO's thinking on the prior art presented to it during the reexamination proceedings.

Because Classified did not file its Counterclaim until February 19, 2008 (D.I. 48), began written discovery only a few days ago, and because trial is not scheduled until nearly one year from now, Classified would not be prejudiced by a stay. Indeed, the filing of the reexamination request and of this motion have occurred much earlier in the life-cycle of a patent litigation than in many cases in which a stay was granted. Accordingly, Inovis' motion to stay the case should be granted.

STATEMENT OF FACTS

Inovis has recently discovered prior art that was not cited to the examiner by the alleged inventors of the 669 Patent during prosecution of that patent. On March 20, 2008, Inovis notified Classified that it intended to file a request for reexamination with the PTO based on this newly discovered prior art. (Ex. A). On April 14, 2008, Inovis filed a request for reexamination of all claims of the 669 Patent. (Ex. B). This reexamination is based on six new pieces of prior art.

The Patent Statute permits any person to request the PTO to reexamine an issued patent. 35 U.S.C. § 302. Within three months of the filing date of a request for reexamination, the PTO must determine if the request raises "a substantial new question of patentability affecting any claim of the patent" and, if the request does, the PTO must grant the reexamination. 35 U.S.C. §§ 303(a), 304. During reexamination, the PTO limits its reexamination to validity issues based upon printed publications and issued patents. 35 U.S.C. § 301. Patent owners may amend the patent claims or add new claims, so long as they do not enlarge the original claim scope. 35 U.S.C. § 305. At the conclusion of the reexamination proceeding, the PTO issues a reexamination certificate canceling any claim determined to be unpatentable, confirming patentable claims, if any, and incorporating any amended or new claims. 35 U.S.C. § 307.

The intent of a reexamination procedure "is to 'start over' in the PTO . . . and to *re*examine the [original] claims, and to *examine* new or amended claims, as they would have been considered if they had been originally examined in light of all the prior art of record in the reexamination proceeding." *Ethicon, Inc. v. Quigg*, 849 F.2d 1422, 1427 (Fed. Cir. 1988) (citations omitted; emphasis in original).

According to the PTO, 2,511 *ex parte* reexamination requests were filed between fiscal years 2002 and 2007, of which 2,389 determinations on these requests were made, and in 2,284 of those requests—or 95.6%—the PTO granted the request for reexamination. (Ex. C, "USPTO Performance and Accountability Report for Fiscal Year 2007" Table 13(a), available at http://www.uspto.gov/web/offices/com/annual/2007/50313a_table13a.html). In 2007, the PTO granted about 97% of the *ex parte* reexamination requests on which determinations were made. (*Id.*). There is therefore a very high probability that Inovis' reexamination requests will be granted and that the PTO will review the patentability of the 669 Patent claims.

ARGUMENT

A district court has the inherent power and broad discretion to stay a patent infringement action pending reexamination proceedings. *Ethicon*, 849 F.2d at 1426-27; *Abbott Diabetes Care, Inc. v. DexCom, Inc.*, C.A. No. 06-514 GMS, 2007 U.S. Dist. LEXIS 73198, at *12 (D. Del. Sept. 30, 2007). In exercising this authority, courts usually stay a patent litigation pending the decision of the PTO in a reexamination proceeding. *See, e.g., Abbott*, 2007 U.S. Dist. LEXIS 73198, at *12-17 (granting motion to stay proceeding pending reexamination of patents-in-suit); *eSoft, Inc. v. Blue Coat Sys., Inc.*, 505 F. Supp. 2d 784, 786 (D. Col. 2007) ("Courts frequently note that '[t]he legislative history surrounding the establishment of the reexamination proceeding evinces congressional approval of district courts liberally granting stays.'") (citations omitted); *Abbott Diabetes Care, Inc. v. DexCom, Inc.*, C.A. No. 05-590 GMS, 2006 U.S. Dist. LEXIS 57469, at *17-21 (D. Del. Aug. 16, 2006) (granting motion to stay proceeding pending reexamination of patents-in-suit); *Echostar Techs. Corp. v. TiVo, Inc.*, No. 5:05-CV-81 (DF), 2006 U.S. Dist. LEXIS 48431 (E.D. Tex. July 14, 2006) (granting stay where reexamination requests were filed before a *Markman* and summary judgment ruling); *Middleton, Inc. v. Minnesota Mining & Mfg Co.*, No. 4:03-cv-40493, 2004 U.S. Dist. LEXIS 16812 (S.D. Iowa Oct. 24, 2004); *Alloc, Inc. v. Unilin Decor N.V.*, 03-253-GMS, 2003 U.S. Dist. LEXIS 11917 (D. Del. July 11, 2003); *Pegasus Dev. Corp. v. DirectTV, Inc.*, 00-1020-GMS, 2003 U.S. Dist. LEXIS 8052 (D. Del. May 14, 2003) (granting motion to stay); *Gioello*, 2001 U.S. Dist. LEXIS 26158; *Rohm and Haas Co. v. Brotech Corp.*, No. 90-109-SLR, 1992 U.S. Dist. LEXIS 3252 (D. Del. 1992).

Courts are guided by the following factors in determining whether a stay is appropriate: "(i) whether a stay would unduly prejudice or present a clear tactical disadvantage to the non-moving party; (ii) whether a stay will simplify the issues in question and trial of the case;

and (iii) whether discovery is complete and whether a trial date has been set." *Abbott*, 2007 U.S. Dist. LEXIS 73198, at *13 (citations omitted).

I. Because Classified Only Asserted Its Counterclaim Two Months Ago And Only Began Discovery Last Week, A Stay Would Not Unduly Prejudice Or Present A Clear Tactical Disadvantage To It

Although Classified accused certain Inovis customers of allegedly infringing the 669 Patent as early as the summer of 2006, Classified has never moved for a preliminary injunction. Classified has therefore implicitly admitted that it is not being irreparably harmed by Inovis' alleged infringement and that it can be adequately compensated by money damages. Moreover, Classified delayed filing its Counterclaim for patent infringement, initially filing a motion to dismiss for lack of subject matter jurisdiction and then, after discovery was conducted on that motion, abruptly withdrawing it before the Court could rule. (D.I. 45). After that delay of Classified's own making, it finally filed a counterclaim on February 19, 2008. (D.I. 48). Classified then did nothing to begin discovery for over two months. Indeed, it was not until last week that Classified finally sent out its first set of document requests and interrogatories relating to patent infringement. (D.I. 63). Because discovery has only just started and because this case is in its infancy, Classified itself would undoubtedly save substantial resources in terms of the time and fees required to conduct discovery, expert discovery, summary judgment briefing, the *Markman* hearing, a trial, post-trial motions, and an appeal. If the PTO invalidates the 669 Patent, Classified will have saved all of that time and expense. But if one or more claims survive reexamation, Classified will still benefit from the PTO's review and analysis of the prior art submitted by Inovis. *Gioello*, 2001 U.S. Dist. LEXIS 26158, at *2-3.

II. A Stay Would Greatly Simplify The Issues, Promote Settlement, And, If Trial Is Necessary, Shorten It

The second factor—whether a stay will simplify the issues in question and trial of the case—weighs greatly in favor of granting the stay. "'One purpose of the reexamination procedure is to eliminate trial of that issue (when the claim is canceled) or to facilitate trial of that issue by providing the district court with the expert view of the PTO (when a claim survives the reexamination proceeding).'" *Abbott*, 2007 U.S. Dist. LEXIS 73198, at *15 (quoting *Gould v. Control Laser Corp.* 705 F.2d 1340, 1342 (Fed. Cir. 1983)). A stay will present an opportunity for the Court to consider the PTO's views concerning the validity of the 669 Patent. Within three months, the PTO will make a formal determination whether the prior art relied upon by Inovis raises a substantial new question of patentability with respect to some or all of the 50 asserted claims of the 669 Patent under *KSR*. *See KSR Int'l Co. v. Teleflex Inc.*, 127 S. Ct. 1727, 1734 (2007). The reasons presented by the PTO for the grant of a reexamination request may provide valuable insight as to the validity of the asserted patent claims. The Court can consider the PTO's views in considering any further proceedings in this action.

> The reexamination process could also result in other numerous efficiencies:
>
> (1) many discovery problems relating to the prior art may be alleviated; (2) the record of the reexamination likely would be entered at trial, reducing the complexity and length of the litigation; (3) the issues, defenses, and evidence will be more easily limited in pre-trial conferences following a reexamination; (4) the outcome of the reexamination process may encourage a settlement without further involvement of the court; and (5) if the patent is declared invalid, the suit likely will be dismissed as to that patent. These efficiencies will result in a reduced cost of litigation for the parties and more effective utilization of the limited resources of the court.

Pegasus, 2003 U.S. Dist. LEXIS 8052, at *5-6 (internal citations omitted). Many issues in the litigation could become moot if the PTO determines that one or more of the claims of the 669 Patent are invalid. *See Abbott*, 2006 U.S. Dist. LEXIS 57469, at *19 ("[I]f the PTO determines

that some or all of the claims of the four patents undergoing reexamination are invalid, then many of the issues in the litigation will become moot."); *Pegasus Dev. Corp*, 2003 U.S. Dist. LEXIS 8052, at *6 ("[A] stay may result in a simplification or reduction of issues for the court's consideration, or it may dispense with the litigation entirely.").

Finally, by staying the case, "the court would not run the risk of inconsistent rulings or issuing advisory opinions." *Abbott*, 2006 U.S. Dist. LEXIS 57469, at *20. The PTO cannot stay a reexamination once the request has been granted, thus, "the court's issuance of a stay is the only way to avoid the potential for conflict." *Gioello*, 2001 U.S. Dist. LEXIS 26158, at *5. Granting a stay "'will maximize the likelihood that neither the Court [sic] nor the parties expend their assets addressing invalid claims.'" *Id.* at *4 (quoting *Softview Computer Prods. Corp. v. Haworth, Inc.*, No. 97 Civ. 8815 KMW HBP, 2000 U.S. Dist. LEXIS 11274, 2000 WL 1134471, at *3 (S.D.N.Y. Aug. 10, 2000)).

III. The Benefits Of The Proposed Stay Are Not Outweighed By The Stage Of This Case

The third factor, whether discovery is complete and whether a trial date has been set, does not outweigh the benefits that will flow from the proposed stay. In fact, this factor greatly favors granting a stay. As noted, although Classified threatened Inovis' customers with patent infringement as early as August 2006, it did not file a claim for infringement until February 19, 2008. (D.I. 48). Classified only began written discovery last week, and no depositions related to Classified's patent-infringement counterclaim have been conducted. Fact discovery is not due to close until August 8, 2008. (D.I. 25; January 15, 2008 Order). The *Markman* hearing is not scheduled until June 24, 2008. (*Id.*). Trial is not scheduled to start until March 23, 2009. (*Id.*). Because discovery is in the very earliest stage and trial is scheduled more than eleven months from now, the benefits discussed above are not even close to being

outweighed by the stage of the case. *See Abbott*, 2006 U.S. Dist. LEXIS 57469, at *20 (granting stay when fact discovery was scheduled to close six months after date of order and trial was scheduled to begin more than one year after date of order); *Pegasus*, 2003 U.S. Dist. LEXIS 8052, at *7 (in granting stay, "the court note[d] that discovery is not complete, and the trial, although scheduled, is some nine months in the future."); *Gioello*, 2001 U.S. Dist. LEXIS 26158, at *1-4 (stay granted where there were pending summary judgment motions on invalidity and infringement; defendant waited 17 months to file for reexamination; trial scheduled only 2 ½ months after decision granting stay).

Indeed, this case is at a much earlier stage than other cases in which a stay has been granted. For example, in *Pegasus*, the case had been pending for two and a half years before the request for reexamination was filed, and summary judgment briefing on invalidity and non-infringement had already been submitted by the parties. *Pegusus*, 2003 U.S. Dist. LEXIS 8052, at *2, 4. In *Gioello*, the defendant did not file its request for reexamination until 17 months after the complaint was filed, and until motions for summary judgment on noninfringement and invalidity had been filed. *Gioello*, 2001 U.S. Dist. LEXIS 26158, at *1, 4. Inovis has acted much more promptly than the defendants in these cases. Accordingly, Classified will suffer no prejudice—much less "undue prejudice"—and the savings to the parties and the Court will be much greater.

CONCLUSION

Granting a stay will conserve judicial resources, the parties' resources, avoid inconsistent rulings, and promote settlement. Accordingly, Inovis respectfully requests that the Court stay this case until the PTO determines whether to grant Inovis' request for reexamination of the 669 Patent and, if the PTO grants the request, until the PTO completes its reexamination.

MORRIS, NICHOLS, ARSHT & TUNNELL LLP

/s/ Julia Heaney

Jack B. Blumenfeld, Esquire (#1014)
Julia Heaney, Esquire (#3052)
1201 North Market Street
P.O. Box 1347
Wilmington, DE 19899-1347
(302) 658-9200
jblumenfeld@mnat.com
jheaney@mnat.com
Attorneys for Plaintiff Inovis USA, Inc.

OF COUNSEL:

David J. Wolfsohn
Erich M. Falke
Jordan L. Jonas
WOODCOCK WASHBURN LLP
Cira Centre, 12th Floor
2929 Arch Street
Philadelphia, PA 19104
(215) 568-3100

Dated: April 28, 2008
2308741

CERTIFICATE OF SERVICE

I, the undersigned, hereby certify that on 28th day of April, 2008, I electronically filed the foregoing with the Clerk of the Court using CM/ECF which will send notification of such filing to the following:

M. Duncan Grant, Esquire
PEPPER HAMILTON LLP

and that copies were caused to be served upon the following individuals in the manner indicated:

ELECTRONIC MAIL

M. Duncan Grant, Esquire
PEPPER HAMILTON LLP
1313 North Market Street, Suite 5100
P. O. Box 1790
Wilmington, DE 19899-1790

Michael A. Lee
Vineet Bhatia, Esquire
Susman Godfrey LLP
1000 Louisiana Street, Suite 5100
Houston, TX 77002-5096

/s/ Julia Heaney

Julia Heaney (#3052)
jheaney@mnat.com

EXHIBIT B-2

UNITED NATIONS

E

Economic and Social Council

Distr.
RESTRICTED

TRADE/WP.4/R.1026/Add.2
22 February 1994

ENGLISH ONLY

ECONOMIC COMMISSION FOR EUROPE

COMMITTEE ON THE DEVELOPMENT OF TRADE

Working Party on Facilitation of
International Trade Procedures
(Item 7 of the provisional agenda of
the Meetings of Experts on Data Elements
and Automatic Data Interchange (GE.1)
Forty-ninth session, 15-16 March 1994)

EDIFACT SECURITY IMPLEMENTATION GUIDELINES

* * *

Submitted by the Rapporteur for the West European EDIFACT Board *

* The present document is reproduced in the form in which it was received by the secretariat.

GE.94- 30557

UN/EDIFACT SECURITY JOINT WORKING GROUP

EDIFACT SECURITY IMPLEMENTATION GUIDELINES

PARIS, NOVEMBER 3rd 1993

CONTENTS

1. SCOPE

This document "EDIFACT Security Implementation Guidelines" gives the technical details needed to implement EDIFACT security, according to the principles outlined in the document entitled "Recommendations for UN/EDIFACT message level security from the UN/EDIFACT Security Joint Working Group".

This document presents the segments and data elements related to security and includes the codes and values specifically needed by security.

It contains examples of message security. These examples are based on EDIFACT payment orders as described in the MIG handbook of Financial messages published by SWIFT, with similar presentation rules. However, the security mechanisms described here are totally independent of the type of message and may be applied to any EDIFACT message.

Note :

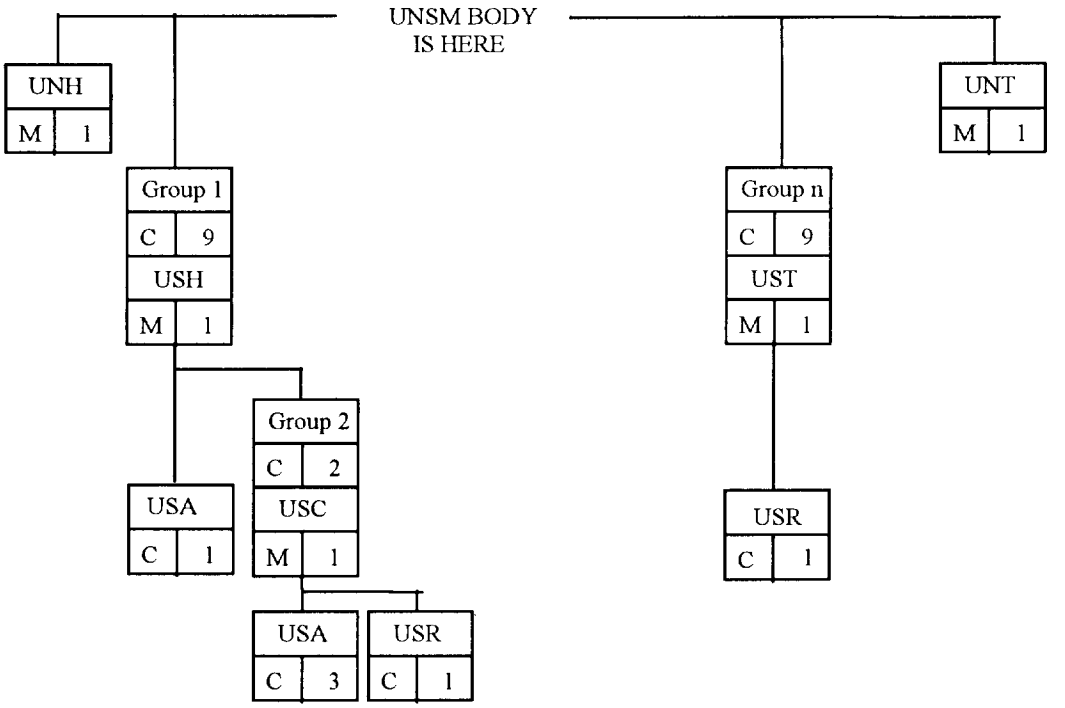
The security service segments described herein are expected to be included in a future version of UN/EDIFACT syntax (ISO 9735). As an interim solution, and for trial use, any user who wishes to incorporate them in EDIFACT messages complying with ISO 9735 (dated 1988), may do so. In this case the composite data element Syntax identifier (S001) of the UNB (Interchange Header) of the interchange containing these messages must contain "RTOx" instead of "UNOx" ("RTOA" instead of "UNOA", for instance).

2. FORMAT SPECIFICATIONS

2.1 Segment Listing

SECTION	TAG	DATA SEGMENT NAME	M/C	NO OF REP
SG 1-1		Segment Group 1	C	9
3.1.1	USH	Security Header	M	1
3.2.1	USA	Security Algorithm	C	1
SG 2-1		Segment Group 2	C	2
3.3.1	USC	Certificate	M	1
3.4.1	USA	Security Algorithm	C	3
3.5.1	USR	Security Result	C	1
SG 3-1		Segment Group n	C	9
3.6.1	UST	Security Trailer	M	1
3.7.1	USR	Security Result	C	1

2.2 Branching Diagram



The general philosophy behind this design is as follows:

- Unless public key techniques are used, Segment Group 2 is omitted.

- The USA segment in Segment Group 1 is related to the actual security service directly applied to the message. For example, for digital signature, it indicates the message-dependent hash function to be used.

- When public keys are used, at least one Segment Group 2, triggered by USC segment, is always required, even if it only contains a reference to a public key in a database at the receiver. Within the USC segment the three USA segments are for the (certified) user public key algorithm, the Certification Authority (CA) public key algorithm and the CA hash function used to calculate the certificate. As soon as the user public key algorithm is present, the Segment Group 2 must include the USR segment.

- The UST segment is used to separate the security trailer from the UNSM body.

3. DATA SEGMENT SPECIFICATION

SEGMENT GROUP 1 (Conditional, 9)

USH	Security Header	M	1
USA	Security Algorithm	C	1
	Segment Group 2	C	2
USC	Certificate	M	1

This segment group identifies the security mechanism applied to the message in which the group is included. It includes the identification of the parties involved in the security process (security elements originator and security elements recipient), the reference of the secured message which includes the date of creation of the security elements and the identification of the security algorithm used (including the technical parameters needed by the algorithm). If the algorithm used requires certificates they may also be conveyed in segment group 2 as part of the present segment group 1.

The algorithm identified in the USA segment is the algorithm directly applied to the message content, either a symmetric algorithm (for message origin authentication, or integrity) or a hash function (for non-repudiation of origin).

In the case of non-repudiation of origin by means of a digital signature, the asymmetric algorithm used to produce the signature will be identified within the certificate segment if it is present, or will be implicitly known by the receiving party if the certificate is not conveyed in the message. In this latter case the asymmetric algorithm may be decided in the Interchange Agreement.

3.1 USH - SECURITY HEADER (Mandatory, 1)

3.1.1 Segment Format

Number	Description	M/C	Format	Special notes
0552	SECURITY STRUCTURE VERSION NUMBER	M	an..3	
0501	SECURITY FUNCTION, CODED	M	an..3	
0534	SECURITY RESULT LINK	M	n2	
0541	SCOPE OF SECURITY APPLICATION, CODED	C	an..3	
0503	RESPONSE TYPE, CODED	C	an..3	
0505	FILTER FUNCTION, CODED	C	an..3	
0507	CHARACTER SET ENCODING, CODED	C	an..3	
0509	ROLE OF SECURITY PROVIDER, CODED	C	an..3	
S500	SECURITY IDENTIFICATION DETAILS	C		
0577	Security party qualifier	M	an..3	
0538	Key name	C	an..35	
0511	Party Id identification	C	an..17	
0513	Code list qualifier	C	an..3	
0515	Code list responsible agency	C	an..3	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
S500	SECURITY IDENTIFICATION DETAILS	C		
0577	Security party qualifier	M	an..3	
0538	Key name	C	an..35	
0511	Party Id identification	C	an..17	
0513	Code list qualifier	C	an..3	
0515	Code list responsible agency	C	an..3	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
0516	SECURITY REFERENCE NUMBER	C	an..35	
S501	SECURITY DATE AND TIME	C		
0517	Date and time qualifier, coded	M	an..3	
0502	Date	C	n8	
0504	Time	C	n6	
0506	UTC offset	C	an..5	

3.1.2 Segment Description

The SECURITY HEADER segment specifies the security service applied to the message in which the segment is included.

If this segment is used, at least SECURITY STRUCTURE VERSION NUMBER (0552), SECURITY FUNCTION (0501) and SECURITY RESULT LINK (0534) **must** be present.

The SECURITY IDENTIFICATION DETAILS composite segments in group 1 either :

- must be used if symmetric methods are used (see 3.2.2), or
- may be used if asymmetric methods are used, and if there is a need to distinguish between the security originator certificate and the security recipient certificate

3.1.3 Segment Rules

There may be several different USH segments within the same message, if different security functions are applied to the message (e. g. integrity and non-repudiation of origin) or if the same security function is simultaneously applied by several entities.

The SECURITY RESULT LINK (0534) will be used to establish a link between one USH segment and the related USR segment.

3.1.3.1 SECURITY STRUCTURE VERSION NUMBER (0552)

It specifies the version number of the format of the security headers and trailers identified by year and status of the UN/EDIFACT service segment directory.

3.1.3.2 SECURITY FUNCTION, CODED (0501)

It specifies the security function applied to the message. One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	NRO	Non-repudiation of origin	The message includes a digital signature protecting the receiver of the message from the sender's denial of having sent the message.
2	AUT	Message origin authentication	The actual sender of the message cannot claim to be some other (authorized) entity.
3	INT	Integrity	The message content is protected against the modification of data.

The security functions are described in the document "Recommendations for UN/EDIFACT message level security from the UN/EDIFACT Security Joint Working Group".

3.1.3.3 SECURITY RESULT LINK (0534)

Contains a number which links a particular USH segment with its corresponding UST segment. The value used is arbitrarily assigned but, within one message, the same value **must** not be used more than once.

3.1.3.4 SCOPE OF SECURITY APPLICATION, CODED (0541)

It specifies the scope of application of the security service defined in the present header, thus it defines the data that have to be taken into account by the related cryptographic process.

This scope may be either :

- the **current Security Header segment group** (from the first character, namely a "U", to the separator ending this Security Header segment group, both included), and the **UNSM body** (from the first character following the separator ending the last Security Header segment group to the separator preceding the first character of the first Security Trailer segment group, both included). In this case any other Security Header or Security Trailer segment group will not be encompassed within this scope. The code used in this case is "SHM",

or

- from the first character (namely a "U", included) of the **current Security Header segment group** to the first character of the **related Security Trailer segment group** (namely a "U", included). The relation between the Security Header and Security Trailer segment groups is provided by the data elements security result link of the USH and of the UST segments. In this case, the scope of application of the security mechanism of the current header will encompass the UNSM body and all the Security

Headers and Security Trailers embedded within this Security Header and its related Security Trailer. The code to use in this case is "SHT",

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	SHM	Security Header and Message body	see explanation above
2	SHT	From Security Header to Security Trailer	see explanation above

3.1.3.5 RESPONSE TYPE, CODED (0503)

It specifies whether a secure acknowledgment from the message recipient is required or not. If it is required, the message sender will expect an AUTACK message to be sent back by the current message recipient to the current message sender, containing this acknowledgement.

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	NA	No acknowledgement required	No AUTACK acknowledgment message expected
2	AC	Acknowledgement required	AUTACK acknowledgment message expected

3.1.3.6 FILTER FUNCTION, CODED (0505)

Identification of the filtering function used for validation results and keys .

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	NUL	No filter	Self explanatory
2	HEX	Hexadecimal filter	Hexadecimal filter
3	ASC	ISO 646 filter	ASCII filter as described in DIS 10126-1 (see note)
4	BAU	ISO 646 Baudot filter	Baudot filter as described in DIS 10126-1
5	EDA	UN/EDIFACT level A filter	UN/EDIFACT level A filter function as described in Annex A of the present document
999	ZZZ	Mutually agreed	Self explanatory

note :
This filtering function is mentioned here for completeness, but does not comply with requirements of UN/EDIFACT level A or B syntax.

3.1.3.7 CHARACTER SET ENCODING, CODED (0507)

Identifies the character set in which the message was coded when security mechanisms were applied.

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	AS7	ASCII 7 bit	Self explanatory
2	AS8	ASCII 8 bit	Self explanatory
3	EBC	EBCDIC IBM 360	IBM 360 machine EBCDIC code
4	IPC	ASCII IBM PC	IBM PC ASCII code
5	VAX	ASCII DEC VAX	DEC VAX ASCII 8 bit code
999	ZZZ	Mutually agreed	Self explanatory

3.1.3.8 ROLE OF SECURITY PROVIDER, CODED (0509)

Identifies the function of the security provider as to the secured item.

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	ISS	Issuer	The security provider is the rightful issuer of the signed document
2	NOT	Notary	The security provider acts as a notary in relation to the signed document
3	CON	Contracting party	The security provider endorses the content of the signed document
4	WIT	Witness	The security provider is a witness, but is not responsible for the content of the signed document
999	ZZZ	Undefined	The role of the security provider is not defined

note :

when this data element is not used, the value "ISS" is assumed.

3.1.3.9 SECURITY IDENTIFICATION DETAILS (S500)

Identification of parties involved in the security process. Two occurrences of this composite data element are possible : one for the security originator, one for the security recipient.

If asymmetric algorithms are used, party identification is performed by the use of certificates. Hence, this composite should be used either :

- if symmetric algorithms are used, or

- if asymmetric algorithms are used and when two certificates are present, in order to distinguish between the originator and the recipient certificates

If these composite data elements are used at least the Security party qualifier (identifying security originator or security recipient) **must** be present.

SECURITY IDENTIFICATION DETAILS composite data elements should be used only if the parties involved in security are not un-ambiguously identified by certificates (use of symmetric methods or asymmetric methods, to clarify the use of more than one certificate).

The identification of the SECURITY IDENTIFICATION DETAILS composite data elements related to the security originator and of the one related to the security recipient is achieved by the Security party qualifier.

If asymmetric methods are used, this SECURITY IDENTIFICATION DETAILS composite data element may be used to identify that a certificate is used by an entity acting as security originator or security recipient. In this case, the identification of the party of the SECURITY IDENTIFICATION DETAILS composite data element (any of the data elements 0511, 0513, 0515, 0586) in the USH segment group will be the same as the identification of the party, qualified as "certificate owner" in the USC segment group, and the security party qualifier will identify the function (originator or recipient) of the party involved.

Security party qualifier (0577)

Specification of the function of the security party identified. One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	MS	Message sender	Identifies the party which generates the security parameters of the message (i.e. security originator).
2	MR	Message receiver	Identifies the party which verifies the security parameters of the message (i.e. security recipient).

Key name (0538)

Identification of a key.

The USA segment allows all the information related to the cryptographic mechanism applied to the secured message to be conveyed.This includes the identification of the key, in the case of symmetric algorithms.

If there is no need to convey a USA segment in the secured message (because the cryptographic mechanisms have been agreed previously between the partners), the data element key name (0538) of the USH segment may be used to establish the key relationship between the sending and receiving parties.

Nevertheless, it is stongly reommended to use either the data element key name (0538) in the USH segment, or the data element algorithm parameter value (0532) with the appropriate qualifier in the USA (data element algorithm parameter qualifier, with the code value "KYN"), but not both of them, within the same message Security Header.

Party Id identification (0511)

Code identifying a party involved in the security process, according to a defined registry of security parties.

Code list qualifier (0513)

Code identifying the type of identification used to register the security parties.

Code list responsible agency (0515)

Code identifying the agency in charge of registration of the security parties.

Party name (0586)

Identification of the security party. Three occurences may be used to allow a complete identification.

3.1.3.10 SECURITY REFERENCE NUMBER (0516)

Identification of the message to which security is applied. This identification is security related and may differ from the identification of the message that may appear elsewhere. The reference number may be used to provide message sequence integrity.

3.1.3.11 SECURITY DATE AND TIME (S501)

Security timestamp of the message to which security is applied. This timestamp is security related and may differ from any dates and times that may appear somewhere else in the message. It may be used to provide message sequence integrity.

Date and time qualifier (0517)

Specification of the type of date and time. The following code **must** be used :

Code	Mnemo.	Meaning	Description
1	STS	Security Timestamp	Security timestamp of the secured messsage

Date (0502)

Specification of the date. Its format **must** be YYYYMMDD (century included).

Time (0504)

Specification of the time. Its format **must** be HHMMSS (HH being in 24 hour clock format).

UTC offset (0506)

Offset from UTC standard time. Its format may be either :

XHHMM (X being P for plus or M for minus, followed by the offset, in hours and minutes), or

XY (X being : C = Central, E = Eastern, M = Mountain, P = Pacific, and Y being : D = daylight time, S = Standard time, T = Time).

3.2 USA - SECURITY ALGORITHM

3.2.1 Segment Format

Number	Description	M/C	Format	Special notes
S502	SECURITY ALGORITHM	M		
0523	Use of algorithm, coded	M	an..3	
0525	Cryptographic mode of operation, coded	C	an..3	
0533	Mode of operation code list identifier	C	an..3	
0527	Algorithm, coded	C	an..3	
0529	Algorithm code list identifier	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier	C	an..3	

3.2.2 Segment Description

This segment is used to identify an algorithm, the technical usage made of it, and to contain the technical parameters required.

The use of algorithm data element specifies the usage made of the algorithm, the algorithm data element identifies the algorithm itself, and the cryptographic mode of operation specifies the mode of operation used.

The mode of operation code list identifier and the algorithm code list identifier identify in which code list the codes of the preceding data elements (mode of operation or algorithm) are defined.

The algorithm parameter composite data element provides space for one parameter. It may be repeated up to 5 times. The number of repetitions actually used will depend on the algorithm used. The order of the parameters is arbitrary but, in each case, the actual value is followed by a coded algorithm parameter Qualifier. Most algorithms in use today will not require parameter values to be the full allowable length.

Where the USA segment is used within a USH segment, the algorithm may be either symmetric, or a hash function.

Asymmetric algorithms shall not be referred to directly in USH segments but may appear only within Segment Group 2, triggered by a USC segment.

3.2.3 Segment Rules

3.2.3.1 SECURITY ALGORITHM (S502)

Use of algorithm, coded (0523)

Specifies the usage made of the algorithm identified by the algorithm data element (0527).

The use of algorithm is required to enable the different SECURITY ALGORITHM composite data elements, used in the different USA segments of Segment Group 2, to be distinguished from each other. In a USH segment the usage of the algorithm is determined by the Security Function qualifier of the USH segment. The use of algorithm **must** be used with one of the following codes :

Code	Mnemo.	Meaning	Description
1	OHA	Owner hashing	Specifies that the algorithm is used by the message sender to compute the hash function on the message. (as in the case of Non-repudiation of Origin identified in the security function qualifier of USH).
2	OSY	Owner symmetric	Specifies that the algorithm is used by the message sender either for integrity or message origin authentication (specified by Security function qualifier in USH).

Cryptographic mode of operation, coded (0525)

Identifies the mode of operation used, for the algorithm specified by the algorithm (0527) data element.
In the USH segment one of the following codes **must** be used :

Code	Meaning	Description
0	NUL	Mode of operation meaningless for the current algorithm.
1	ECB	DES modes of operation, Electronic Code Book; FIPS Pub 81 (1981); ANSI X3.106; IS 8372 (64 bits); ISO 10116 (n-bits).
2	CBC	DES modes of operation, Cipher Block Chaining; FIPS Pub 81 (1981); ANSI X3.106; IS 8372 (64 bits); ISO 10116 (n-bits).
3	CFB1	DES modes of operation, Cipher feedback; FIPS Pub 81 (1981); ANSI X3.106; IS 8372 (64 bits); ISO 10116 (n-bits).
4	CFB8	DES modes of operation, Cipher feedback; FIPS Pub 81 (1981); ANSI X3.106; IS 8372 (64 bits); ISO 10116 (n-bits).
5	OFB	DES modes of operation. FIPS Pub 81 (1981); IS 8372 (64 bits); ISO 10116 (n-bits).
6	MAC	Message Authentication Code ISO 8731-1, using DES CBC mode.
7	DIM1	Data integrity mechanism using a cryptographic check function; ISO DIS 9797, first method
8	DIM2	Data integrity mechanism using a cryptographic check function; ISO DIS 9797, second method
9	MDC2	Modification Detection Code - IBM System Journal, vol 30, no 2, 1991.
10	HDS1	Hash functions - Part 2 : Hash functions using a n-bit block cipher algorithm providing a single length hash code. ISO CD 10118-2.
11	HDS2	Hash functions - Part 2 : Hash functions using a n-bit block cipher algorithm providing a double length hash code. ISO CD 10118-2.
12	SQM	Square-mod n hash function for RSA. Annex D, CCITT X 509. ISO 9594-8.
13	NVB7.1	Dutch Standard hash function for banking.
14	NVBAK	Dutch Banking Standard, NVB Authenticity Mark, published by the NVB, May 1992.
999	ZZZ	Mutually agreed.

Mode of operation code list identifier (0533)

Specification of the code lists used to identify the cryptographic mode of operation. When the codes defined above, by the UN/EDIFACT SJWG, as published in the present document, are used the value "1" **must** be used.

Algorithm, coded (0527)

Identifies the algorithm. In the USH segment one of the following codes **must** be used :

Code	Meaning	Description
1	DES	Data Encryption Standard. FIPS Pub 46 (January 1977).
2	MAA	Message Authentication Algorithm. Banking-Approved Algorithms for message Authentication. ISO 8731-2.
3	FEAL	FEAL Fast Data Encipherment Algorithm.
4	IDEA	International Data Encryption Algorithm : Lai X., Massey J. "A Proposal for a New Block Encryption Standard", Proceedings of Eurocrypt'90, LNCS vol 473, Springer-Verlag, Berlin 1991, and Lai X., Massey J. "Markov Ciphers and Differential Cryptanalysis", Proceedings of Eurocrypt'91, LNCS vol 547, Springer-Verlag, Berlin 1991.
5	MD4	The MD4 Message digest algorithm. Rivest R. RSA Data Security Inc. (1990).
6	MD5	The MD5 Message digest algorithm. Rivest R. Dusse S. RSA Data Security Inc. (1991).
7	RIPEMD	Extension of the MD4 - Ripe Report CS - R9324, April 93.
8	SHA	Secure Hashing Algorithm.
9	AR/DFP	Hash function of the German banking industry, submitted to ISO/IEC JTC 1/SC 27/WG 2, Doc N179.
999	ZZZ	Mutually agreed.

The presence of a particular algorithm in the list does not imply any endorsement of that algorithm.

The above algorithms are those currently in common use (November 1993). It is expected that new algorithms will be added as they become available, and are generally accepted.

Algorithm code list identifier (0529)

Specification of the code lists used to identify the algorithm. When the codes defined above by the UN/EDIFACT SJWG, as published in the present document, are used the value "1" **must** be used.

3.2.3.2 ALGORITHM PARAMETER (S503)

Algorithm parameter value (0532)

This component contains the value of a parameter required by the algorithm referenced in the algorithm data element. The precise type, usage and format of the value is specified in the immediately following algorithm parameter qualifier (0531). If necessary, this value is filtered by the filter function identified in the FILTER FUNCTION, CODED data element (0505) of the USH segment (key names do not need to be filtered).

Algorithm parameter qualifier (0531)

Identifies the type of the algorithm parameter value that immediately precedes it.

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	IVC	Initalisation Value, cleartext	Identifies the algorithm parameter value as an unencrypted initialisation value.
2	IVE	Initialisation Value, encrypted under a symmetric key	Identifies the algorithm parameter value as an initialisation value which is encrypted under the symmetric data key.
3	IVP	Initialisation Value, encrypted under a public key	Identifies the algorithm parameter value as an initialisation value encrypted under the public key of the receiving entity.
4	IVZ	Initialisation Value, format mutually agreed	Identifies the algorithm parameter value as an initialisation value in a format agreed between the two parties.
5	KYE	Symmetric key, encrypted under a symmetric key	Identifies the algorithm parameter value as a symmetric key which is encrypted with a previously agreed algorithm under a previously exchanged symmetric key.
6	KYP	Symmetric key, encrypted under a public key	Identifies the algorithm parameter value as a symmetric key encrypted under the public key of the receiving entity.
7	KYS	Symmetric key, signed and encrypted	Identifies the algorithm parameter value as a symmetric key signed under the sender's secret key, then encrypted under the receiver's public key.
8	KYD	Symmetric key encrypted under an asymmetric key common to the sender and the receiver	Identifies the algorithm parameter value as a symmetric key encrypted under an asymmetric key common to the sender and the receiver (use of Diffie and Hellmann scheme, for instance).
9	KYN	Symmetric key name	Identifies the algorithm parameter value as the name of a symmetric key. This may be used in the case where a key relationship has already been established between the sender and receiver.
10	KKN	Key encrypting key name	Identifies the parameter value as the name of a key encrypting key.
11	KYZ	Symmetric key, format mutually agreed	Identifies the algorithm parameter value as a symmetric key in a format agreed between the two parties.
999	ZZZ	Parameter value is mutually agreed	Identifies the algorithm parameter value as having a usage and format that is mutually agreed.

SEGMENT GROUP 2 (Conditional, 2)

USC	Certificate	M	1
USA	Security Algorithm	C	3
USR	Security Result	C	1

When asymmetric algorithms are used, this segment group contains the data necessary to validate the security methods applied to a message.

In its most common form, the certificate will include the public key and the credentials of the certificate owner signed by the Certificate Issuer.

A certificate may either be conveyed completely (the USC segment, 3 USA segments and the USR segment), or may be conveyed only as a Certificate Identifier (the USC segment identifying the Certificate), to refer to a public key that is already known by the entities involved or retrieved from a data base.

Two occurrences of this Segment group are allowed, one being the message sender certificate (that the message receiver will use to verify the message sender's signature), the other being the message receiver certificate (presumably only referred to by Certificate Reference) in the case where the receiver public key is used by the sender for confidentiality of symmetric keys.

If two certificates (sender's and receiver's) are simultaneously present within one security header, the Security Identification Details data element (S500) together with the Certificate Reference data element (0536) allow them to be differentiated.

3.3 USC - CERTIFICATE (Mandatory, 1)

3.3.1 Segment Format

Number	Description	M/C	Format	Special notes
0536	CERTIFICATE REFERENCE	C	an..35	
S500	SECURITY IDENTIFICATION DETAILS	C		
0577	Security party qualifier	M	an..3	
0538	Key name	C	an..35	
0511	Party Id identification	C	an..17	
0513	Code list qualifier	C	an..3	
0515	Code list responsible agency	C	an..3	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
S500	SECURITY IDENTIFICATION DETAILS	C		
0577	Security party qualifier	M	an..3	
0538	Key name	C	an..35	
0511	Party Id identification	C	an..17	
0513	Code list qualifier	C	an..3	
0515	Code list responsible agency	C	an..3	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
0544	FORMAT CERTIFICATE VERSION	C	an..3	
0505	FILTER FUNCTION, CODED	C	an..3	
0507	CHARACTER SET ENCODING, CODED	C	an..3	
0543	CHARACTER SET REPERTOIRE, CODED	C	an..3	
0546	USER AUTHORISATION LEVELS	C	an..35	
S505	SEPARATOR FOR SIGNATURE	C		
0548	Separator for signature	C	an..4	
0551	Separator for signature qualifier	C	an..3	
S505	SEPARATOR FOR SIGNATURE	C		
0548	Separator for signature	C	an..4	
0551	Separator for signature qualifier	C	an..3	
S505	SEPARATOR FOR SIGNATURE	C		
0548	Separator for signature	C	an..4	
0551	Separator for signature qualifier	C	an..3	
S505	SEPARATOR FOR SIGNATURE	C		
0548	Separator for signature	C	an..4	
0551	Separator for signature qualifier	C	an..3	
S501	SECURITY DATE AND TIME	C		
0515	Date and time qualifier, coded	M	an..3	
0502	Date	C	n8	
0504	Time	C	n6	
0506	UTC offset	C	an..5	
S501	SECURITY DATE AND TIME	C		
0515	Date and time qualifier, coded	M	an..3	
0502	Date	C	n8	
0504	Time	C	n6	
0506	UTC offset	C	an..5	
S501	SECURITY DATE AND TIME	C		
0515	Date and time qualifier, coded	M	an..3	
0502	Date	C	n8	
0504	Time	C	n6	
0506	UTC offset	C	an..5	

3.3.2 Segment Description

USC segments will be needed when public key cryptography is used, even if certificates are not used. Either the full certificate is present (including the USR segment), or the only data

elements of the certificate are the Certificate reference (0536) and the SECURITY IDENTIFICATION DETAILS (S500) composite data element identifying the isuer Certification Authority or the SECURITY IDENTIFICATION DETAILS (S500) composite data element identifying the Certificate Owner, including its public key name. The presence of a full certificate may be avoided if the certificate has already been exchanged by the two parties.

3.3.3 Segment Rules

3.3.3.1 CERTIFICATE REFERENCE (0536)

Uniquely identifies one certificate for a Certification Authority. This field may be used to refer to a certificate when the whole certificate is not conveyed. The unique certificate reference may be obtained by the Certificate reference (0536) and the SECURITY IDENTIFICATION DETAILS (S500) composite data element identifying the Certification Authority.

3.3.3.2 SECURITY IDENTIFICATION DETAILS (S500)

Identification of parties involved in the certification process.

Two occurrences of this composite data element are possible : one for the Certificate Owner (identifying the entity which signs the message), one for the Certificate Issuer (Certification Authority or CA).

When this composite data element is used at least the Security Party Qualifier **must** be present.

The identification of Certificate Owner and Certification Authority is achieved by the data element Security Party Qualifier.

Security party qualifier (0577)

Specification of the function of the security party identified. One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
3	OW	Certificate Owner	Identifies the party which owns the Certificate.
4	AX	Authenticating party	Party which certifies that the document (i. e. the certificate) is authentic.

Key name (0538)

Identification of a public key : either the public key of the owner of this certificate, or the public key related to the secret key used by the Certificate Issuer (CA) to sign this certificate. In the latter case, this field allows a Certification Authority to use several keys, either for separate applications, or consecutive generations of CA keys.

Party Id identification (0511)

Code identifying a party involved in the security process, according to a defined registry of security parties. In the USC segment this party may be either :

- the party which owns the Certificate (Certificate Owner), or
- the party which certifies that the document (i. e. the certificate) is authentic (Authenticating party : CA)

Code list qualifier (0513)

Code identifying the type of identification used to register the security parties.

Code list responsible agency (0515)

Code identifying the agency in charge of registration of the security parties.

Party name (0586)

Identification of the security party. Three occurences may be used to allow a complete identification.

3.3.3.3 FORMAT CERTIFICATE VERSION (0544)

Version number of the version of the certificate, identified by year and status of the UN/EDIFACT service segment directory.

3.3.3.4 FILTER FUNCTION, CODED (0505)

Identification of the filtering function used for validation results and keys, within the certificate.

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	NUL	No filter	Self explanatory
2	HEX	Hexadecimal filter	Hexadecimal filter
3	ASC	ISO 646 filter	ASCII filter as described in DIS 10126-1 (see note)
4	BAU	ISO 646 Baudot filter	Baudot filter as described in DIS 10126-1
5	EDA	UN/EDIFACT level A filter	UN/EDIFACT level A filter function as described in Annex A of the present document
999	ZZZ	Mutually agreed	Self explanatory

note :

This filtering function is mentioned here for completeness, but does not comply with the requirements of UN/EDIFACT level A or B syntax.

3.3.3.5 CHARACTER SET ENCODING, CODED (0507)

Identifies the character set in which the certificate was coded when security mechanisms were applied to it.

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	AS7	ASCII 7 bit	Self explanatory
2	AS8	ASCII 8 bit	Self explanatory
3	EBC	EBCDIC IBM 360	IBM 360 machine EBCDIC code
4	IPC	ASCII IBM PC	IBM PC ASCII code
5	VAX	ASCII DEC VAX	DEC VAX ASCII 8 bit code
999	ZZZ	Mutually agreed	Self explanatory

3.3.3.6 CHARACTER SET REPERTOIRE, CODED (0543)

Identifies the syntax level used to create the certificate, when security mechanisms were applied to it.

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	UNOA	UN/EDIFACT syntax level A	Self explanatory
2	UNOB	UN/EDIFACT syntax level B	Self explanatory
3	UNOC	UN/EDIFACT syntax level C	Self explanatory
4	UNOD	UN/EDIFACT syntax level D	Self explanatory
5	UNOE	UN/EDIFACT syntax level E	Self explanatory
6	UNOF	UN/EDIFACT syntax level F	Self explanatory

3.3.3.7 USER AUTHORISATION LEVELS (0546)

Specification of the privileges, authorisation level, etc. associated with the owner of the certificate.

3.3.3.8 SEPARATOR FOR SIGNATURE (S505)

Identifies the characters used as syntactical separators between all components or within composite components, of the USC segment when the signature was computed. The syntactical separators used in the message may be different to these characters. The data element Separator for signature qualifier (0551) identifies each separator. If the composite data elements SEPARATOR FOR SIGNATURE (S505) are not present, the syntactical characters used are those currently used in the message.

Separator for signature (0548)

Separator used when the signature was computed. In order to avoid translator problems, this separator is represented by its value in the character set identified by the CHARACTER SET ENCODING data element (0507), hexa-filtered on, at least, two characters. For example the separator "'" is coded "27" (two characters), ":" is coded "3A" (two characters), the release character "?" is coded "3F" (two characters) and the separator "+" is coded "2B", if ASCII 8bit code page is used.

Separator for signature qualifier (0551)

Identifies each separator (either separator between data element or separator within a composite). One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	SEG	Segment separator	Specifies that this is the separator between segments.
2	DAT	Data separator	Specifies that this is the separator between data elements.
3	COM	Composite separator	Specifies that this is the separator within a composite data element.
4	REL	Release character	Specifies that this is the release character.

3.3.3.9 SECURITY DATE AND TIME (S501)

Identification of the dates and times involved in the certification process.

Three occurrences of this composite data element are possible : one for the certificate generation date and time, one for the certificate start of validity period , one for the certificate end of validity period.

The distinction between the certificate generation date and time, the certificate start of validity period and of the certificate end of validity period is achieved by the Date and time qualifier.

Date and time qualifier (0515)

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
2	CGT	Certificate generation time	Identifies the date and time of generation of the certificate by the Certification Authority.
3	CSV	Certificate start of validity period	Identifies the date and time from which the certificate must be considered valid.
4	CEV	Certificate end of validity period	Identifies the date and time until which the certificate must be considered valid.

Date (0502)

Specification of the date. Its format **must** be YYYYMMDD (century included).

Time (0504)

Specification of the time. Its format **must** be HHMMSS (HH being in 24 hour clock format).

UTC offset (0506)

Offset from UTC standard time. Its format may be either :

XHHMM (X being P for plus or M for minus, followed by the offset, in hours and minutes),
or
XY (X being : C = Central, E = Eastern, M = Mountain, P = Pacific, and Y being : D = daylight time, S = Standard time, T = Time).

3.4 USA - SECURITY ALGORITHM (Mandatory, 3)

3.4.1 Segment Format

Number	Description	M/C	Format	Special notes
S502	SECURITY ALGORITHM	M		
0523	Use of algorithm, coded	M	an..3	
0525	Cryptographic mode of operation, coded	C	an..3	
0533	Mode of operation code list identifier	C	an..3	
0527	Algorithm, coded	C	an..3	
0531	Algorithm code list identifier	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier, coded	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier, coded	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier, coded	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier, coded	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier, coded	C	an..3	

3.4.2 Segment Description

This segment is used to identify an algorithm, the technical usage made of it, and to hold the technical parameters required.

The use of algorithm data element specifies the usage made of the algorithm, the algorithm data element identifies the algorithm itself and the cryptographic mode of operation data element specifies the mode of operation used.

The mode of operation code list identifier and the algorithm code list identifier identify the code list in which the codes of the preceding data elements (mode of operation or algorithm) are defined.

The algorithm parameters composite data element provides space for one parameter. It may be repeated up to 5 times. The number actually used will depend on the algorithm used. The order of the parameters is arbitrary but, in each case, the actual value is followed by a coded algorithm parameter qualifier. Most algorithms in use today will not require parameter values to be the full allowable length.

In Segment Group 2, triggered by the USC segment, three USA segments are present. These are :

1. the algorithm used by the Certificate Issuer to compute the hash value of the Certificate (hashing function)

2. the algorithm used by the Certificate Issuer to generate the Certificate (i. e. to sign the result of the hash function computed on the certificate content) (asymmetric algorithm)

3.a - either the algorithm used by the sender to sign the message (i. e. to sign the result of the hash function described in the USH segment, computed on the message content) (asymmetric algorithm),

3.b - or the receiver's asymmetric algorithm used by the sender to encrypt the key required by a symmetric algorithm applied to the message content and refered to by the Segment Group 1 triggered by the USH segment (asymmetric algorithm),

These three occurrences of the USA segment are distinguished by the Use of algorithm data element (0523).

3.4.3 Segment Rules

3.4.3.1 SECURITY ALGORITHM (S502)

Use of algorithm (0523)

Specifies the usage made of the algorithm identified by Algorithm data element (0527). In a USA segment within a USC one of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
3	ISG	Issuer signing	Specifies that the algorithm is used by the Certificate Issuer (CA) to sign the hash result computed on the certificate
4	IHA	Issuer hashing	Specifies that the algorithm is used by the Certificate Issuer (CA) to compute the hash result on the certificate
5	OCF	Owner enciphering	Specifies that the algorithm is used by the message sender to encrypt a symmetric key
6	OSG	Owner signing	Specifies that the algorithm is used by the message sender to sign either the hash result computed on the message or the symmetric keys
7	OCS	Owner enciphering or signing	Specifies that the algorithm is used by the message sender to encrypt a symmetric key or sign the hash result computed on the message

Cryptographic mode of operation, coded (0525)

Identifies the mode of operation used, for the algorithm specified by the algorithm (0527) data element.
In the USC segment one of the following codes **must** be used :

Code	Meaning	Description
0	NUL	Mode of operation meaningless for the current algorithm.
9	MDC2	Modification Detection Code - IBM System Journal, vol 30, no 2, 1991.
10	HDS1	Hash functions - Part 2 : Hash functions using a n-bit block cipher algorithm providing a single length hash code. ISO CD10118-2.
11	HDS2	Hash functions - Part 2 : Hash functions using a n-bit block cipher algorithm providing a double length hash code. ISO CD10118-2.
12	SQM	Square-mod n hash function for RSA. Annex D, CCITT X 509. ISO 9594-8.
15	MCCP	Banking key management by means of asymmetric algorithms, Algorithms using the RSA cryptosystem. Signature construction by means of a separate signature. ISO CD 11166-2.
16	DSMR	Digital Signature Scheme Giving message recovery. ISO 9796.
999	ZZZ	Mutually agreed.

Mode of operation code list identifier (0533)

Specification of the code lists used to identify the cryptographic mode of operation. When the codes defined above by the UN/EDIFACT SJWG, as published in the present document, are used the value "1" **must** be used.

Algorithm, coded (0527)

Specifies the algorithm. In the USC segment one of the following codes **must** be used :

Code	Meaning	Description
1	DES	Data Encryption Standard. FIPS Pub 46 (January 1977).
5	MD4	The MD4 Message digest algorithm. Rivest R. RSA Data Security Inc. (1990).
6	MD5	The MD5 Message digest algorithm. Rivest R. Dusse.S RSA Data Security Inc. (1991).
7	RIPEMD	Extension of the MD4 - Ripe Report CS - R9324, April 93.
8	SHA	Secure Hashing Algorithm.
9	AR/DFP	Hash function of the German banking industry, submitted to ISO/IEC JTC 1/SC 27/WG 2, Doc N179.
10	RSA	Rivest, Shamir, Adleman: A Method for obtaining Digital Signatures and Public Key Cryptosystems. Communications of the ACM, Vol.21(2), pp 120-126 (1978).
11	DSA	Digital Signature Algorithm/Digital Signature Standard NIST Pub 1993 Draft.
12	RAB	Rabin, "Digitalized signatures and public-key functions as intractable as factorization", MIT Laboratory for Computer Science Technical Report LCS/TR-212, Cambridge, Mass, 1979.
999	ZZZ	Mutually agreed.

Algorithm code list identifier (0529)

Specification of the code lists used to identify the algorithm. When the codes defined above by the UN/EDIFACT SJWG, as published in the present document, are used the value "1" **must** be used.

3.4.3.2 ALGORITHM PARAMETER (S503)

Algorithm parameter value (0532)

This component contains the value of a parameter required by the algorithm referenced in the algorithm data element. The precise type, usage and format of the value is specified in the immediately following algorithm parameter qualifier. If necessary, this value is filtered by the filter function identified in the FILTER FUNCTION data element (0505) of the USC segment (key names do not need to be filtered).

Algorithm parameter qualifier (0531)

Identifies the type of the algorithm parameter value that immediately precedes it.

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
12	MOD	Modulus	Identifies the algorithm parameter value as the modulus of a public key which is to be used according to the function defined by the use of algorithm.
13	EXP	Exponent	Identifies the algorithm parameter value as the exponent of a public key which is to be used according to the function defined by the use of algorithm.
14	MLN	Modulus Length	Identifies the algorithm parameter value as the length of the modulus (in bits) of the public key used in the algorithm. The length is independent of whatever filtering function may be in use.
15	PR1	Generic parameter 1	Identifies the algorithm parameter value as the first generic parameter (see note)
16	PR2	Generic parameter 2	Identifies the algorithm parameter value as the second generic parameter (see note)
17	PR3	Generic parameter 3	Identifies the algorithm parameter value as the third generic parameter (see note)
18	PR4	Generic parameter 4	Identifies the algorithm parameter value as the fourth generic parameter (see note)
19	PR5	Generic parameter 5	Identifies the algorithm parameter value as the fifth generic parameter (see note)
20	PR6	Generic parameter 6	Identifies the algorithm parameter value as the sixth generic parameter (see note)
21	PR7	Generic parameter 7	Identifies the algorithm parameter value as the seventh generic parameter (see note)
22	PR8	Generic parameter 8	Identifies the algorithm parameter value as the eighth generic parameter (see note)
23	PR9	Generic parameter 9	Identifies the algorithm parameter value as the ninth generic parameter (see note)
24	PRA	Generic parameter 10	Identifies the algorithm parameter value as the tenth generic parameter (see note)
999	ZZZ	Parameter value is mutually agreed	Identifies the algorithm parameter value as having a usage and format that is mutually agreed.

note :
These generic parameters are provided to allow the use of any algorithm requiring identification of parameters different from the parameters defined above.

When the DSA algorithm (NIST, Pub 1993) is used, PR1 contains parameter "P", PR2 contains parameter "Q", PR3 contains parameter "G", PR4 contains parameter "Y".

3.5 USR - SECURITY RESULT (Mandatory, 1)

3.5.1 Segment Format

Number	Description	M/C	Format	Special notes
S508	VALIDATION RESULT	M		
0560	Validation value	M	an..256	
0560	Validation value	C	an..256	

3.5.2 Segment Description

The USR segment included in the USC segment contains the signature computed by the Certification Authority by signing the hash result computed on the data of the credentials.

In the case of signature algorithms requiring two parameters to express the result, the two data elements validation result are used in an order described in the documents about these algorithms.

3.5.3 Segment Rules

3.5.3.1 VALIDATION RESULT (S508)

Validation value (0560)

This component contains the digital signature computed by the Certification Authority on the data of the credentials. This signature is computed using first the hash function defined by the qualifier "issuer hashing" ("4") in the use of algorithm data element, then the asymmetric algorithm defined by the qualifier "issuer signing" ("3") in the use of algorithm data element.

The digital signature is computed according to the rules and with the parameters specified in the USC segment (CHARACTER SET ENCODING, SEPARATOR FOR SIGNATURE, CHARACTER SET REPERTOIRE). The signature computation starts with the first character of the USC segment (namely a "U") and ends with last character of the last USA segment (including the separator following this USA segment).

This data element is filtered, if necessary, by the filter function identified in the FILTER FUNCTION data element (0505) of the USC segment.

The length of this data element is determined by the length of the key (one of the Algorithm parameter data elements, qualified by the Algorithm parameter qualifier "modulus length" ("14"), of the Issuer signature algorithm) and the filter function applied to the result of the signature process.

In the case of RSA signature, only one validation value data element is used.

In the case of DSA signature two validation value data elements are required. The first one corresponds to the parameter known as "r", the second one to the parameter known as "s".

SEGMENT GROUP n (Conditional, 9)

	Segment Group 3	C	9
UST	Security Trailer	M	1
USR	Security Result	C	1

This segment group contains the link with the related USH segment and the security result corresponding to the security functions specified in this USH segment. For every security trailer (Segment Group n, triggered by UST) there is one corresponding security header (Segment Group 1, triggered by USH).

3.6 UST - SECURITY TRAILER (Mandatory, 1)

3.6.1 Segment Format

Number	Description	M/C	Format	Special notes
0534	SECURITY RESULT LINK	M	n2	

3.6.2 Segment Description

This segment is used to separate the UNSM body from the security trailer.

The UST segment contains a number which links the UST segment with its corresponding USH segment.

3.6.3 Segment Rules

3.6.3.1 SECURITY RESULT LINK (0534)

Contains a number which links a particular USH segment with its corresponding UST segment. The value used is arbitrarily assigned but, within one message, the same value **must** not be used more than once.

3.7 USR - SECURITY RESULT (Conditional, 1)

3.7.1 Segment Format

Number	Description	M/C	Format	Special notes
S508	VALIDATION RESULT	M		
0560	Validation value	M	an..256	
0560	Validation value	C	an..256	

3.7.2 Segment Description

Contains the security result corresponding to the security functions specified in the linked USH segment

In the case of signature algorithms requiring two parameters to express the result, the two data elements validation result are used in an order described in the documents about these algorithms.

3.7.3 Segment Rules

3.7.3.1 VALIDATION RESULT (S508)

Validation value (0560)

Contains the security result corresponding to the security functions specified in the linked USH segment.

This data element is filtered, if necessary, by the filter function identified in the FILTER FUNCTION data element (0505) of the USH segment.

The length of this data element is determined by the length of the key (one of the Algorithm parameter data elements, qualified by the Algorithm parameter qualifier "modulus length" ("14"), of the Owner signature algorithm) and the filter function applied to the result of the signature process.

In the case of RSA signature, only one validation value data element is used.

In the case of DSA signature two validation value data elements are required. The first one corresponds to the parameter known as "r", the second one to the parameter known as "s".

4 HOW TO PROTECT AN EDIFACT MESSAGE

The following are some of the more fundamental steps to be taken in order to implement security for EDIFACT messages. For further details and explanation of principles, refer to Document A, ISO 7498-2 and CCITT X.509.

The first step is to identify (in cooperation with business associates) the need for security services. The security services available in the EDIFACT world are revisited below, and it is important to establish which of these are required in the business relations to prevent the identified threats. Typically, the needs could be defined by the request for auditing, internally as well as externally. The basic security services available at the sender's end are the following:

- message content integrity
- message origin authentication
- non-repudiation of origin

These services are not independent, and it is thus not necessary to additionally include the services which are achieved implicitly. For example, the use of the non-repudiation of origin service implicitly achieves message content integrity.

The following table summarizes these inclusions:

also implies : Use of :	Message Content Integrity	Message Origin Authentication	Non-repudiation of origin
Message Content Integrity	yes		
Message Origin Authentication	yes	yes	
Non-repudiation of origin	yes	yes	yes

Consequently, the sender would choose at most one of the three.

Non-repudiation of receipt is a service to be initiated by the receiver. It could either be requested by the sender in the message or mandated in an interchange agreement. A special message, AUTACK, is being developed to convey the receipt.

4.1 Bilateral agreement/third party

If security services are being integrated, additional agreements of course have to be set up with the business partners. There are a number of different approaches available, of which two extremes are briefly presented here.

A minimal requirement would be a bilateral agreement with each individual partner, agreeing on security services, algorithms, codes, key management methods, actions in case of misconduct, etc. A draft of such an agreement is available from the EEC TEDIS programme. In this case, very little security-related information needs to be included in the message itself.

The other extreme would be to involve a third party acting as a certification authority, which registers all users and issues certificates to certify the users' public keys. In this situation, it may be adequate simply to conclude an agreement with the certification authority. The certification authority would typically be responsible for blacklisting as well. In this case, more comprehensive security-related information may need to be included.

The security services have been integrated into the EDIFACT setting in a manner that offers maximum flexibility, and caters for both extremes described above, as well as for any intermediate situation.

4.2 Practical aspects

There are, of course, a number of different aspects that need to be addressed in order to realize these security services, such as key generation, the need for a translator capable of handling security segments, internal procedures to make full use of the security services, such as storing incoming messages with digital signatures, the use of multiple signatures, etc.

It should be emphasized that integration of security services is completely transparent to, and independent of, the communication protocols used. If a system allows the transmission of an EDIFACT message, it will also allow the transmission of a secured EDIFACT message.

4.3 Procedure for constructing a secured message

First, an EDIFACT Message is created. Then the appropriate security services are determined. If these are based on digital signatures, the persons possessing the secret keys have to be involved, directly or indirectly. This does not have to take place immediately after the generation of the message.
Likewise, on incoming messages, the first step would be to verify the security services , and, just as in the paper world, possibly to store the secured message for later auditing and documentation.

4.4 Application boundaries of security services

The computation of each of the integrity and authentication values and of the digital signatures either :

starts with and includes the associated Message Security Header and the message body itself, and ends with the last character of the message body (it includes the separator preceding the associated Message Security Trailer), or

starts with and includes the associated Message Security Header, all the other embedded Message Security Headers and Trailers, the message body itself, and ends with the last character of the message body, or of the last embedded Message Security Trailer, if any, (it includes the separator preceding the associated Message Security Trailer).

The order in which the security services are performed, is left entirely to the users, as all services, can be completely independent of each other. In particular, if multiple signatures are used, without

embedding of Security Headers and Security Trailers , the order in which they are calculated, and verified, is of no consequence.

4.5 Choice of security techniques

Once the required security services are identified, the technique needed to provide them should be chosen.

The available techniques can be categorized in the following way :

ACRONYM	TECHNIQUE	ALGORITHM CODE (if relevant, as defined in § 3.2.3.1 and 3.4.3.1)
SA	symmetric authentication algorithm	1, 2 or 3
SE	symmetric encipherment algorithm	1 to 4
AS	asymmetric algorithm	10 to 12
HF	hashing function	5 to 9
SN/DT	sequence number or date/timestamp	irrelevant (defined in segment USH)

The security service required can be achieved with either one, or a combination of, the above mentioned techniques.

The table below gives an overview of well-accepted combinations. Crosses (x) on the horizontal entries indicate which technique or (combination of techniques) can be used to achieve the security service. Note that several entries for one security service mean that several combinations of security techniques may be used.

Note that this table only explains the principle of how a security service may be achieved with different types of techniques. It is based on the cryptographic state of the art at the time of writing this document and may be therefore subject to changes.

Whether or not an identified security service is reached depends largely on the exact choice of cryptographic algorithms and of their parameters. In some cases an algorithm used with wrong parameters is not secure enough (RSA with too small modulus). In other cases a combination of algorithms may lead to weaknesses (using some hash-function in combination with RSA, for instance).

The choice of algorithm and parameters should be based on the current state of the art of cryptographic research and hardware developments.

Convention :

Upper case (X) denotes a technique sufficient by itself
Lower case (x) denotes a combination of techniques

Technique Service	SA	SE	AS	HF	SN/DT
Message Content Integrity :					
	X				
	x			x	
	x		x		
		x		x	
			x	x	
Message Origin Authentication :					
	X				
	x			x	
	x		x		
		x		x	
			x	x	
Non-repudiation of Origin :			x	x	
Message Sequence Integrity :					
	x				x
	x			x	x
	x		x		x
		x			x
			x		x

5 MESSAGE PROTECTION EXAMPLES

Three examples are provided herein to illustrate different application of security service segments.

"Example 1 : Message Origin Authentication" shows how security service segments may be used when a **symmetric algorithm** based method is applied, to provide Message Origin Authentication. The symmetric key has been exchanged previously between the partners, and the Security Header group contains only two rather simple segments.

"Example 2 : Non repudiation of Origin, first technique" shows how security service segments may be used when an **asymmetric algorithm** based method is applied, to provide Non repudiation of Origin. The algorithm applied directly to the message is a **hash-function**, which does not require any key exchange between the partners. The hash-value is signed by an asymmetric algorithm. The public key needed by the receiver to verify the signature of the message is included in a certificate segment which is conveyed in the Security Header group of the message. This certificate is signed by its issuer (the "Authority") and contains the public key of the Authority, in order that any partner may verify the integrity and authenticity of the certificate.

"Example 3 : Non repudiation of Origin, second technique" shows how security service segments may be used when an **asymmetric algorithm** based method is applied, to provide Non repudiation of Origin. The algorithm applied directly to the message is a **symmetric algorithm**, which requires a symmetric key exchange between the partners, and provides an "integrity value". This symmetric key is exchanged within the Security Header group of the message, encrypted by means of an asymmetric algorithm, under the public key of the expected receiver.
The integrity value is signed by an asymmetric algorithm. The public key needed by the receiver to verify the signature of the message is included in the first certificate segment which is conveyed in the Security Header group of the message. This certificate is signed by its issuer (the "Authority") and contains the public key of the Authority, in order that any partner may verify the integrity and authenticity of the certificate.
A second certificate segment contains the reference of the public key of the expected receiver, used by the message sender to protect the symmetric key.
This technique is currently used by the french Banks in the ETEBAC 5 system (secured file transfer between banks and corporate customers).

In the last two examples, any partner, trusting the Authority, may verify the signature of the received message using only data contained in the message.

5.1 EXAMPLE 1 : Message Origin Authentication

5.1.1 Narrative

Company A orders Bank A, sort code 603000 to debit its account number 00387806 on April 9th 1994 in the amount of 54345.10 Pounds Sterling. The amount is to be paid to Bank B, sort code 201827, in favour of account number 00663151 of Company B, West Dock, Milford Haven. The payment is in settlement of invoice 62345. The contact name at the Beneficiary is Mr. Jones in the Sales Department.

Bank A requires the payment order to be secured by the security function "Message Origin Authentication".
This is achieved by generating a "Message Authentication Code" (MAC) with the symmetric "Data Encryption Algorithm" (DEA) according to ISO 8731-1 at the message sender's side, which is to be validated by Bank A. It is assumed that the secret DEA-key has previously been exchanged between Company A and Bank A.

Remark :

In the following, only the security relevant parts of the message will be refered to.

5.1.2 Security details

SECURITY HEADER	
SECURITY STRUCTURE VERSION NUMBER	1994 service segment directory
SECURITY FUNCTION	Message Origin Authentication
SECURITY RESULT LINK	The reference of this header is 1
FILTER FUNCTION	All binary values (MAC) are filtered with hexadecimal filter
CHARACTER SET ENCODING	The message was coded in ASCII 8 bits when the MAC was generated.
SECURITY IDENTIFICATION DETAILS Message sender (party which generates the Message Authentication Code).	Mr. SMITH of Company A
SECURITY IDENTIFICATION DETAILS Message receiver (party which verifies the Message Authentication Code).	Bank A
SECURITY REFERENCE NUMBER	The security sequence number of this message is 001
SECURITY DATE AND TIME	The security time stamp is : date : 1994 04 09 time : 13:59:50
SECURITY ALGORITHM	
SECURITY ALGORITHM Use of Algorithm Cryptographic mode of operation Algorithm	A symmetric algorithm is used to achieve Message Origin Authentication. A MAC is computed, according to ISO 8731-1. DES algorithm is used.
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	The key called MAC-KEY1 is used. Identifies the preceding algorithm parameter value as the name of a previously exchanged symmetric key.
SECURITY TRAILER	
SECURITY RESULT LINK	The reference of this trailer is 1
SECURITY RESULT	
VALIDATION RESULT	4 Byte validation result (Message Authentication Code)

5.1.3 PAYORD EDIFACT MESSAGE WITH SECURITY SERVICE OF MESSAGE ORIGIN AUTHENTICATION

No	Data Segment	Message
1	Message Header	UNH+326+PAYORD:1:911:RT'
2	Security Header	USH+94W+2+1+++2+2++1:::::MR.SMITH:COMPANY A+2::::BANK A+1+1:19940409:135950'
3	Security Algorithm	USA+2:6::1+MAC-KEY1:9'
4	Beginning of Message	BGM+450+AZ341234+137:920405:101'
5	Name and Address	NAD+BE++COMPANY B:WEST DOCK: MILFORD HAVEN'
6	Contact Informations	CTA++:MR. JONES, SALES'
7	Financial Institution Information	FII+OR+00387806:COMPANY A+603000:154:132'
8	Financial Institution Information	FII+BF+00663151+201827:154:132'
9	Date/Time/Period	DTM+203:920414:101'
10	Monetary Amount	MOA+7+9:54345,10:GBP'
11	Document/Message Details	DOC+380+62345'
12	Security Trailer	UST+1'
13	Security Result	USR+C1FFA1BC'
14	Message Trailer	UNT+14+326'

The DEA key used for this example is :

01 23 45 67 89 AB CD EF

5.1.4 Explanation

1. The reference for this message is 326.
2. The security stucture version is the current one, the security function required is "Message Origin Authentication" (code 2), the security result link to the trailer is 1, the filter function is Hexadecimal filter (code 2), ASCII 8bit character set encoding is used (code 2), the Message Authentication Code is generated by Mr. Smith, Company A (message sender : code 1) and validated by Bank A (message receiver : code 2). The security reference number of this message is 1, its security date and time (time stamp : code 1) is :
   date : 19940409 time : 13:59:50.
3. A symmetric algorithm (code 2) is used to achieve "Message Origin Authentication", namely a MAC (code 6) is formed with the "Data Encryption Algorithm" (code 1) according to ISO 8731-1. The symmetric secret key for generating and validating the MAC has been previously exchanged between the parties and is refered to by its name "MAC-KEY1".
   The process starts with and includes the first character of the "Security Header" USH segment and ends with the last character preceding the "Security Trailer" segment (UST).
4. Not security related
5. Not security related
6. Not security related
7. Not security related
8. Not security related
9. Not security related
10. Not security related
11. Not security related
12. Security trailer. The reference of this security trailer is 1.
13. 4 byte validation result (Message Authentication Code) according to ISO 8731-1.
    Hex-filtered : 8 bytes.
14. They are 14 segments (including the trailer) in this message number 326.

note : the MAC value in this example has not actually been computed, so it is irrelevant to try to verify it.

5.2 EXAMPLE 2 : Non-repudiation of Origin, first technique

5.2.1 Narrative

Bank A wants the security service of Non repudiation of Origin on the payment order from Company A, performed by Mr. Smith.

The interchange agreement between the parties establishes that the security service of Non Repudiation of Origin, required by Bank A, shall be achieved for payment orders, by Mr. Smith of Company A, with the use of one digital signature.

The certificate identifying the public key of Mr. Smith is issued by an Authority trusted by both parties, the Certificate Issuer.

5.2.2 Security details

SECURITY HEADER	
SECURITY STRUCTURE VERSION NUMBER	1994 service segment directory
SECURITY FUNCTION	Non repudiation of origin
SECURITY RESULT LINK	The reference of this header is 1
RESPONSE TYPE	No acknowledgment required
FILTER FUNCTION	All binary values (signatures) are filtered with hexadecimal filter
CHARACTER SET ENCODING	The message was coded in ASCII 8 bits when its signature was generated.
SECURITY REFERENCE NUMBER	The security sequence number of this message is 202.
SECURITY DATE AND TIME	The security time stamp is : date : 1994 01 15 time : 10:05:30
SECURITY ALGORITHM	Hash function used by Mr. SMITH for signature
SECURITY ALGORITHM Use of Algorithm Cryptographic mode of operation Algorithm	 An Owner hashing algorithm is used. Hash function CD 10118-2 Hash functions using a n- bit block cipher algorithm applied to provide a double length hash code (128 bits); initializing values : IV = 0F 0F 0F 0F 0F 0F 0F 0F IV' = F0 F0 F0 F0 F0 F0 F0 F0; padding rules as first variant paragraph B.3.1 of the standard; transformation u and u' as specified in annex A of the standard. DES block cipher algorithm is used.
CERTIFICATE	Certificate of Mr. SMITH
CERTIFICATE REFERENCE	This Certificate is referenced, by AUTHORITY : 00000001.
SECURITY IDENTIFICATION DETAILS Certificate Owner	 Mr. SMITH of Company A
SECURITY IDENTIFICATION DETAILS Certificate Issuer Key name	 Mr. SMITH's certificate was generated by a Certification Authority called : AUTHORITY. The Public Key of AUTHORITY used to generate Mr. SMITH's certificate is PK1
FORMAT CERTIFICATE VERSION	Version of Certificate of UN/EDIFACT service segment directory 94W.
FILTER FUNCTION	All binary values (keys and digital signatures) are filtered with hexadecimal filter
CHARACTER SET ENCODING	The credentials of the certificate were coded in ASCII 8 bits when the certificate was generated.
SEPARATOR CHARACTER FOR SIGNATURE	The separator characters used when signature was computed were "'" between segments, "+" between data elements, ":" within composite data elements and the release character was "?".
SECURITY DATE AND TIME Date and time	Certificate generation time Mr. SMITH certificate was generated on 931215 at 14:12:00
SECURITY DATE AND TIME Date and time	Certificate start of validity period Validity period of Mr. SMITH's starts : 1994 01 01 000000
SECURITY DATE AND TIME Date and time	Certificate end of validity period Validity ends of Mr. SMITH's starts : 1994 12 31 235959

SECURITY ALGORITHM	Asymmetric algorithm used by Mr. SMITH to sign
SECURITY ALGORITHM Use of Algorithm Cryptographic mode of operation Algorithm	 An Owner signing algorithm is used. No mode of operation is relevant here. RSA is the asymmetric algorithm.
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 Mr SMITH's public key Identifies the preceding algorithm parameter as a Public exponent for signature verification
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 Mr SMITH's modulus Identifies the preceding algorithm parameter as a Modulus for signature verification
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 Mr SMITH's modulus is 512 bit long Identifies the preceding algorithm parameter as the length of Mr SMITH's modulus (in bits)
SECURITY ALGORITHM	Hash function used by AUTHORITY to generate Mr SMITH's certificate
SECURITY ALGORITHM Use of Algorithm Cryptographic mode of operation Algorithm	 An Issuer hashing algorithm is used. Hash function CD 10118-2 Hash functions using a n- bit block cipher algorithm applied to provide a double length hash code (128 bits); initializing values : IV = 0F 0F 0F 0F 0F 0F 0F 0F IV' = F0 F0 F0 F0 F0 F0 F0 F0; padding rules as first variant paragraph B.3.1 of the standard; transformation u and u' as specified in annex A of the standard. DES block cipher algorithm is used.
SECURITY ALGORITHM	Asymmetric algorithm used by AUTHORITY to sign
SECURITY ALGORITHM Use of Algorithm Cryptographic mode of operation Algorithm	 An Issuer signing algorithm is used. No mode of operation is relevant here. RSA is the asymmetric algorithm.
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 AUTHORITY's public key Identifies the preceding algorithm parameter as a Public exponent for signature verification
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 AUTHORITY's modulus Identifies the preceding algorithm parameter as a Modulus for signature verification
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 AUTHORITY's modulus is 512 bit long Identifies the preceding algorithm parameter as the length of AUTHORITY's modulus (in bits)
SECURITY RESULT	Digital signature of the Certificate
VALIDATION RESULT	512 Bit digital signature
SECURITY TRAILER	
SECURITY RESULT LINK	The reference of this security trailer is 1
SECURITY RESULT	Digital signature of the message
VALIDATION RESULT	512 Bit digital signature

5.2.3 PAYORD EDIFACT MESSAGE WITH SECURITY SERVICE OF NON REPUDIATION OF ORIGIN

No	Data Segment	Message
1	Message Header	UNH+326+PAYORD:1:911:RT'
2	Security Header	USH+94W+1+1+++1+2+2+++++202+1:19940115:100530'
3	Security Algorithm	USA+1:11::1'
4	Certificate	CER+00000001+3:::::MR.SMITH:COMPANYA+4:PKI::::AUTHORITY+94W+2+2+++27:1+2B:2+3A:3+3F:4+2:19931215:141200+3:19940101:000000+4:19941231:235959'
5	Security Algorithm	USA+6:0::10+00010001:13+CA056F9C89708200E822A8A19BC6ADD430807705DE2D5AFA7934F63EA8E7C280379C02DA758799F34F2C0D1C2747F98E43A1EAA4BE8195FC24A17BE70446F95F:12+0512:14'
6	Security Algorithm	USA+4:11::1'
7	Security Algorithm	USA+3:0::10+00010001:13+FC5959DE40BF4DFB13930E7FF703FB6089864535F328981CE1CFDF43A010DB79955CA8171FC5D463A488C5E5227E8F9BDD562EE4E23BD3F29827A53233596341:12+0512:14'
8	Security Result	USR+C80209FC7BEF4EAF589CA5366DC609B5F1729C8FE3A56A314108E3C1620B0C0C2E42F007A227A7809783262CB9AE5717B4096695FCC774F3FCCB8E8D6D2BE0AE'
9	Beginning of Message	BGM+450+AZ341234+137:920405:101'
10	Name and Address	NAD+BE++COMPANY B:WEST DOCK:MILFORD HAVEN'
11	Contact Informations	CTA++:MR.JONES, SALES'
12	Financial Institution Information	FII+OR+00387806:COMPANY A+603000:154:132'
13	Financial Institution Information	FII+BF+00663151+201827:154:132'
14	Date/Time/Period	DTM+203:920414:101'
15	Monetary Amount	MOA+7+9:54345,10:GBP'
16	Document/Message Details	DOC+380+62345'
17	Security Trailer	UST+1'
18	Security Result	USR+6BD7DC037A28325075D51B22A1EF46FB651CD3244A86D7C4130E8EB7F7A9CC6EBF750A7172478F2033715C44662DC7C212F734B5AB814719717A758F2C04E1A3'
19	Message Trailer	UNT+19+326'

5.2.4 Explanation

1. The reference for this message is 326.
2. The security stucture version is the current one, the security function required is "Non repudiation of Origin" (code 1), the security result link to the trailer is 1, the filter function is Hexadecimal filter (code 2), ASCII 8bit character set encoding is used (code 2). The security reference number of this message is 202, its security date and time (time stamp : code 1) is :
   date :1994 01 15
   time : 10:05:30.
3. The message sender (Mr. SMITH) uses hash function CD 10118-2.
4. This certificate is referenced 00000001. Certificate Owner is Mr. SMITH of Company A. Mr SMITH's certificate have been generated by AUTHORITY. AUTHORITY uses public key PK1. Version 94W of Certificate format is used, it is hexa-filtered and ASCII coded, and the certificate signature was computed with the separators explicitly specified. This certificate was issued on Dec 15 1993, and it is valid between Jan 1st 1994 and Dec 31rd 1994.
5. Mr SMITH is signing with RSA 512 bit, its public key is 65537 (decimal) or 010001 (hexadecimal), the modulus being :

   CA 05 6F 9C 89 70 82 00 - E8 22 A8 A1 9B C6 AD D4
   30 80 77 05 DE 2D 5A FA - 79 34 F6 3E A8 E7 C2 80
   37 9C 02 DA 75 87 99 F3 - 4F 2C 0D 1C 27 47 F9 8E
   43 A1 EA A4 BE 81 95 FC - 24 A1 7B E7 04 46 F9 5F

6. AUTHORITY hash function is CD 10118-2.
7. AUTHORITY is signing the certificate with RSA 512 bit, its public exponent is 65537 (decimal) or 010001 (hexadecimal), the modulus being :

   FC 59 59 DE 40 BF 4D FB - 13 93 0E 7F F7 03 FB 60
   89 86 45 35 F3 28 98 1C - E1 CF DF 43 A0 10 DB 79
   95 5C A8 17 1F C5 D4 63 - A4 88 C5 E5 22 7E 8F 9B
   DD 56 2E E4 E2 3B D3 F2 - 98 27 A5 32 33 59 63 41

8. The Certificate signature.
9. Not security related
10. Not security related
11. Not security related
12. Not security related
13. Not security related
14. Not security related
15. Not security related
16. Not security related
17. Security trailer. The reference of this security trailer is 1.
18. The payment order signature.
19. They are 19 segments (including the trailer) in this message number 326.

note : the MAC value in this example has not actually been computed, so it is irrelevant to try to verify it.

5.3 EXAMPLE 3 : Non-repudiation of Origin, second technique

5.3.1 Narrative

Bank A wants the security service of Non repudiation of Origin on the payment order from Company A, performed by Mr. Smith. Company A requests a secured acknowlegment by Bank A (non repudiation of receipt) which will be conveyed in a AUTACK message.

The interchange agreement between the parties establishes that the security service of Non Repudiation of Origin shall be achieved for payment orders with the use of one digital signature. Both parties agree that this signature is computed by a 512 bit RSA (asymmetric algorithm) upon a 64 bit-integrity value computed by a CBC mode DES (symmetric algorithm).The certificate identifying the public key of Mr. Smith is issued by an Authority trusted by both parties.

5.3.2 Security details

SECURITY HEADER	
SECURITY STRUCTURE VERSION NUMBER	1994 service segment directory
SECURITY FUNCTION	Non repudiation of origin
SECURITY RESULT LINK	The reference of this header is 1
RESPONSE TYPE	Acknowledgment required
FILTER FUNCTION	All binary values (signatures) are filtered by hexadecimal filter
CHARACTER SET ENCODING	The message was coded in ASCII 8 bits when its signature was generated.
SECURITY IDENTIFICATION DETAILS Message sender (party securing the message).	Mr. SMITH of Company A
SECURITY IDENTIFICATION DETAILS Message receiver (party verifying message security).	Bank A
SECURITY REFERENCE NUMBER	The security sequence number of this message is 001.
SECURITY DATE AND TIME	The security time stamp is : date : 1994 01 15 time : 10:05:30
SECURITY ALGORITHM	Symmetric algorithm used to compute an integrity value.
SECURITY ALGORITHM Use of Algorithm Cryptographic mode of operation Algorithm	 An Owner hashing algorithm is used. Cipher Block Chaining; ISO 10116 (n-bits). A 64-bit integrity value is computed; initialization value is binary zero; a DEA secret-key is used. It is transmitted encrypted under Bank A public key. DES block cipher algorithm is used.
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 Symmetric key encrypted under Bank A public key. Identifies the preceding algorithm parameter value as a symmetric key encrypted under a public key
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 Cleartext initialisation value (all binary 0's). Identifies the preceding algorithm parameter value as a cleartext initialisation value
CERTIFICATE	Certificate of Mr. SMITH (message sender)
CERTIFICATE REFERENCE	This Certificate is referenced : 00000001, by AUTHORITY.
SECURITY IDENTIFICATION DETAILS Certificate Owner	 Mr. SMITH of Company A
SECURITY IDENTIFICATION DETAILS Certificate Issuer Key name	 Mr. SMITH's certificate was generated by a Certification Authority called : AUTHORITY. The Public Key of AUTHORITY used to generate Mr. SMITH's certificate is PK1
FORMAT CERTIFICATE VERSION	Version of Certificate of UN/EDIFACT service segment directory 94W.
FILTER FUNCTION	All binary values (keys and digital signatures) are filtered with hexadecimal filter
CHARACTER SET ENCODING	The credentials of the certificate were coded in ASCII 8 bits when the certificate was generated.
SEPARATOR CHARACTER FOR SIGNATURE	The separator characters used when signature was computed were "'" between segments, "+" between data elements, ":" within composite data elements and the release character was "?".

SECURITY DATE AND TIME Date and time	Certificate generation time Mr. SMITH certificate was generated on 931215 at 14:12:00
SECURITY DATE AND TIME Date and time	Certificate start of validity period Validity period of Mr. SMITH's starts : 1994 01 01 000000
SECURITY DATE AND TIME Date and time	Certificate end of validity period Validity period of Mr. SMITH's ends : 1994 12 31 235959
SECURITY ALGORITHM	Asymmetric algorithm used by Mr. SMITH to sign
SECURITY ALGORITHM Use of Algorithm Cryptographic mode of operation Algorithm	 An Owner signing algorithm is used. No mode of operation is relevant here. RSA is the asymmetric algorithm.
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 Mr SMITH's public key Identifies the preceding algorithm parameter as a Public exponent for signature verification
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 Mr SMITH's modulus Identifies the preceding algorithm parameter as a Modulus for signature verification
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 Mr SMITH's modulus is 512 bit long Identifies the preceding algorithm parameter as the length of Mr SMITH's modulus (in bits)
SECURITY ALGORITHM	Hash function used by AUTHORITY to generate Mr SMITH's certificate
SECURITY ALGORITHM Use of Algorithm Cryptographic mode of operation Algorithm	 An Issuer hashing algorithm is used. Square-mod n hash function for RSA. Annex D, CCITT X509. ISO 9594-8. RSA asymmetric algorithm.
SECURITY ALGORITHM	Asymmetric algorithm is used by AUTHORITY to sign
SECURITY ALGORITHM Use of Algorithm Cryptographic mode of operation Algorithm	 An Issuer signing algorithm is used. No mode of operation is relevant here. RSA is the asymmetric algorithm.
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 AUTHORITY's public key Identifies the preceding algorithm parameter as a Public exponent for signature verification
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 AUTHORITY's modulus Identifies the preceding algorithm parameter as a Modulus for signature verification
ALGORITHM PARAMETER Algorithm parameter value Algorithm parameter qualifier	 AUTHORITY's modulus is 512 bit long Identifies the preceding algorithm parameter as the length of AUTHORITY's modulus (in bits)
SECURITY RESULT	Digital signature of the Certificate
VALIDATION RESULT	512 Bit digital signature
CERTIFICATE	Certificate of Bank A (message receiver)
CERTIFICATE REFERENCE	Bank A's public key related to certificate referenced 00001001 is used
SECURITY TRAILER	
SECURITY RESULT LINK	The reference of this security trailer is 1
SECURITY RESULT	Digital signature of the message
VALIDATION RESULT	512 Bit digital signature

5.3.3 PAYORD EDIFACT MESSAGE WITH SECURITY SERVICE OF NON REPUDIATION OF ORIGIN

No	Data Segment	Message
1	Message Header	UNH+326+PAYORD:1:911:RT'
2	Security Header	USH+94W+1+1++2+2+2++1:::::MR.SMITH :COMPANY A+2:::::BANK A+326+1:19940115:100530'
3	Security Algorithm	USA+1:2::1+6E97569DB22F444EA70BAC31A08 AD5CFB7C955F2FE0545047F490E2C44FDF33E31 E4CAF54EDA66622EF96239A8FCF9B269457FEE CC5CC8BBE8AF689B9D29C011:6+000000000000 0000:1''
4	Certificate	CER+00000001+3:::::MR.SMITH:COMPANYA +4:PK1::::AUTHORITY+94W+2+2+++27:1:2 B:2:3A:3:3F:4+2:19931215:141200+3:19940101:0 00000+4:19941231:235959'
5	Security Algorithm	USA+6:0::10+00000003:13+F496F058AB9DCB7 9491C0A12E5A966259FAAB40202E26793032E34A 8D30252F105A78F20DEB376AA1403E302371A82 37F737F2A4CDE0227F0E9A62275275BEF7:12+0 512:14'
6	Security Algorithm	USA+4:12::10'
7	Security Algorithm	USA+3:0::10+00000003:13+B684FFC6DCC4656 53E6D2D4E167B8101D28A6266A984D815303E36 0E204D10C2C3D2F807500FA2082F364A9F8B4DB 79E55571468D945E5EA92C29022392A1E8D:12+0 512:14'
8	Security Result	USR+06792DD06115CA63D8A21C05C76D4C131 7E6BED21BFBC5119FEFC9FA84FA92B919793A2 C8F939F88073E9C53F798CAD0A58FFD8C03208F C7439D75EAB543060E'
9	Certificate	CER+00001001+3:::::BANK A+4:::::AUTHORITY'
10	Beginning of Message	BGM+450+AZ341234+137:920405:101'
11	Name and Address	NAD+BE++COMPANY B:WEST DOCK: MILFORD HAVEN'
12	Contact Informations	CTA++:MR.JONES, SALES'
13	Financial Institution Information	FII+OR+00387806:COMPANY A+603000:154:132'
14	Financial Institution Information	FII+BF+00663151+201827:154:132'
15	Date/Time/Period	DTM+203:920414:101'
16	Monetary Amount	MOA+7+9:54345,10:GBP'
17	Document/Message Details	DOC+380+62345'
18	Security Trailer	UST+1'
19	Security Result	USR+B531AE86CA202000DC4653B625CA547503 52907E7C613DD524A1847A9D4B792BF47FCA768 F822D701DD653DCF6ED5AC8CA2C152E45735E 82C1910F7B026018CE'
20	Message Trailer	UNT+20+326'

5.3.4 Explanation

1. The reference for this message is 326.
2. The security stucture version is the current one, the security function required is "Non repudiation of Origin" (code 1), the security result link to the trailer is 1, an acknowledgment is required, the filter function is Hexadecimal filter (code 2), ASCII 8bit character set encoding is used (code 2). The message sender is Mr. SMITH of COMPANY A, and the expected message receiver is Bank A. The security reference number of this message is 001, its security date and time (time stamp : code 1) is :
   date :1994 01 15, time : 10:05:30.
3. The message sender (Mr. SMITH) uses DEA CBC mode integrity result as hash code of the message. The DEA secret-key used is : 01 23 45 67 89 AB CD EF. This DEA-secret key is encrypted under a public RSA key : the modulus used is :

   CA 05 6F 9C 89 70 82 00 - E8 22 A8 A1 9B C6 AD D4
   30 80 77 05 DE 2D 5A FA - 79 34 F6 3E A8 E7 C2 80
   37 9C 02 DA 75 87 99 F3 - 4F 2C 0D 1C 27 47 F9 8E
   43 A1 EA A4 BE 81 95 FC - 24 A1 7B E7 04 46 F9 5F

   the public exponent is Fermat4 = 010001 hexadecimal (65537 decimal)
   The DEA algorithm uses a nul initialisation value.
4. This certificate is referenced 00000001. Certificate Owner is Mr. SMITH of Company A. Mr SMITH's certificate have been generated by AUTHORITY. AUTHORITY uses public key PK1. Version 94W of Certificate format is used, it is hexa-filtered and ASCII coded, and the certificate signature was computed with the separators explicitly specified. This certificate was issued on Dec 15 1993, and it is valid between Jan 1st 1994 and Dec 31rd 1994.
5. Mr SMITH is signing with RSA 512 bit, its public key is 3 (decimal), the modulus being :

   F4 96 F0 58 AB 9D CB 79 - 49 1C 0A 12 E5 A9 66 25
   9F AA B4 02 02 E2 67 93 - 03 2E 34 A8 D3 02 52 F1
   05 A7 8F 20 DE B3 76 AA - 14 03 E3 02 37 1A 82 37
   F7 37 F2 A4 CD E0 22 7F - 0E 9A 62 27 52 75 BE F7

6. AUTHORITY hash function is square-mod n, annex D CCITT X509, ISO 9594-8. The modulus used is the signature modulus of AUTHORITY (given in 7. below).
7. AUTHORITY is signing the certificate with RSA 512 bit, its public key : 3 (decimal), modulus :

   B6 84 FF C6 DC C4 65 65 - 3E 6D 2D 4E 16 7B 81 01
   D2 8A 62 66 A9 84 D8 15 - 30 3E 36 0E 20 4D 10 C2
   C3 D2 F8 07 50 0F A2 08 - 2F 36 4A 9F 8B 4D B7 9E
   55 57 14 68 D9 45 E5 EA - 92 C2 90 22 39 2A 1E 8D

8. The Certificate signature.
9. Bank A's certificate related to the public key used to encrypt the DEA secret-key (see 3. above) is referenced 00001001. It is issued by AUTHORITY.
10. Not security related
11. Not security related
12. Not security related
13. Not security related
14. Not security related
15. Not security related
16. Not security related
17. Not security related
18. Security trailer. The reference of this security trailer is 1.
19. The payment order signature.
20. They are 20 segments (including the trailer) in this message number 326

note : the cryptographic values in this example have not actually been computed, so it is irrelevant to try to verify it.

ANNEX A : PROPOSAL FOR AN UN/EDIFACT LEVEL A FILTER FUNCTION

1 Rationale

Hexadecimal filtering doubles the number of characters required to represent binary data. This is a waste of space. Other existing and standardized filter functions are either not adequate for UN/EDIFACT syntax levels A and B (ISO 646) because they map to almost the full printable ISO set (94 out of the 96 printable characters), or because they are not really more space-performant than hexadecimal filtering (the BAUdot filter).
It is thus advisable to define a filter function which is sufficiently simple and which maps to (a subset of) the UN/EDIFACT level A character set, while being more efficient than the hexadecimal filter.

2 UN/EDIFACT character sets

The level A character set possesses 44 characters whose use is without restrictions. In addition to those 44, 4 separator characters and 8 characters not allowed for TELEX transmissions are part of the set.

All those characters are also part of the UN/EDIFACT level B character set, which is not intended at all for TELEX transmission, and which possesses 82 normal characters and 3 not-printable separators.

3 3 by 2 filtering.

To represent 2 binary characters by 3 filtered characters a minimum of 41 characters are required in the set : indeed 41 ** 3 = 68 921 > 65 536 > 64 000 = 40 ** 3

4. The EDA filter.

Having 44 allowed characters, let us avoid use of the space character part of those 44 and to filter every pair of input characters (if odd, filter only the last character in 2 resulting ones) by :

- considering the binary value of the unsigned integer formed by the pair of characters (this value depends naturally on the LITTLE_ENDIAN / BIG_ ENDIAN (either Least or Most Significant Byte first) nature of the computer in use. Standardize for BIG_ENDIANs : first byte most significant)
- represent the value by a succession of 3 numbers (2 for last odd byte), in the range 0 to 42, which are :
  - the result of the division by 1849 (43 squared) (absent for last odd byte)
  - the value modulo 1849 divided by 43,
  - the value modulo 43.
- to map each number in the UN/EDIFACT level A alphabet by the correspondence table :

0 to 9	represent	0 to 9
A to Z	represent	10 to 35
( ) , - . / =	represent	36 up to 42 in the given order.

5 Defiltering

To defilter : map each of the 43 characters back to its value between 0 and 42,
if at least 3 filtered characters remained, compute : c1 * 1849 + c2 *43 + c3 = short integer
else at least 2 remained and compute : c1 * 43 + c2 = character value.

Remarks :
a. The short integer result should be < 65 536
b. The character result should be < 256
c. In a LITTLE_ENDIAN computer, switch the 2 characters of the short integer result.

EXHIBIT B-3

UNITED
NATIONS

E

Economic and Social
Council

Distr.
RESTRICTED

TRADE/WP.4/R.1026/Add.3
23 February 1994

ENGLISH ONLY

ECONOMIC COMMISSION FOR EUROPE

COMMITTEE ON THE DEVELOPMENT OF TRADE

Working Party on Facilitation of
International Trade Procedures
(Item 7 of the provisional agenda
of the Meeting Of Experts on Data Elements
and Automatic Data Interchange (GE.1)
- Forty-ninth session, 15-16 March 1994)

AUTACK: SECURE AUTHENTIFICATION AND ACKNOWLEDGEMENT MESSAGE

Development of United Nations Standard Messages (UNSMs)

MESSAGE TYPE SUBMITTED AS STATUS 1 FOR INFORMATION

Submitted by the Security Joint Working Group *

* * *

* The present document is reproduced in the form in which it was received by the secretariat.

GE.94- 30556

UN/EDIFACT

DRAFT RECOMMENDATION

Secure authentification and acknowledgement message

This message is available for formal trial for a period of at least twelve months from the date of approval by the UN/ECE/TRADE/WP.4.

Organizations are invited to trial this message and are requested to notify their Rapporteur Team Secretariat of their intention. Comments on the results from the trial should also be forwarded to the Secretariat as soon as they are available. Based on the results of the trials, a UNSM will be issued.

The segments, composite and simple data elements and codes for the trial of this message are contained in the Trial directory (status 1). However, this material may differ from that in the TDID directory (status 2) having the same identifying tags. Any differences will be reconciled prior to the message becoming a UNSM.

Message Type	: AUTACK
Version	: 1
Release	: 1
Contr. Agency	: UN
Status	: 1
Date	: 94-03

SOURCE: Security Joint Working Group

Message type specification, AUTACK

CONTENTS

Secure authentication and acknowledgement message

For general information on UN standard message types see UN Trade Data Interchange Directory, UNTDID, Part 4, Section 2.5, UN/ECE UNSM General Introduction.

0. Introduction

This specification provides the definition of the Secure Authentication and Acknowledgement Message (AUTACK), to be used in Electronic Data Interchange (EDI) between partners involved in administration, commerce and transport.

1. Scope

1.1 Functional Definition

The purpose of the message is to provide security services for messages sent separately.

1.2 Field of Application

This message can be applied for both national and international trade. It is based on universal practice and is not dependent on the type of business or industry. It can be used for any combination of messages that need to be secured between two parties.

1.3 Principles

A secure authentication and acknowledgement message used as an authentication message is sent by the originator of messages and interchanges sent separately or by a party having authority to act on behalf of the originator, to facilitate message origin authentication, validation of integrity of content, validation of message sequence integrity or non-repudiation of origin of these messages.

A secure authentication and acknowledgement message used as an acknowledgement message is sent by the recipient of previously received messages or by a party having authority to act on behalf of the recipient, to facilitate confirmation of receipt, validation of integrity of content, validation of completeness or non-repudiation of receipt of these messages.

This AUTACK message can apply to one or more messages from one or more interchanges or to one or more interchanges.

The security services are provided by cryptographic mechanisms applied to the content of the original messages or interchanges. The results of these mechanisms form the body of the AUTACK message, supplemented by the relevant data such as the reference to the cryptographic methods used, the reference number and the date and time of the original entities.

The AUTACK message is completed by the standard security header and trailer groups.

1.3.1 Use for authentication and non-repudiation of origin.

AUTACK used for authentication and non-repudiation of origin, as announced in its USB segment, can be used in two major ways. In either way, security errors do not apply, but a validation result is always required.

1.3.1.1 Hashing of references and secured AUTACK.

The secured entity (message or interchange) is referenced in an occurrence of the USX segment. Underneath it, at least one occurrence of USY is required referring to a header group of the present AUTACK thanks to the use of a "0534 Security result link" data-element. This particular header only contains the description of a hashing method, which is the one applied to the referred entity from its beginning service segment upto and including its ending service segment end. In the applicable header, the data-element "501 Security function; coded" refers to "referenced entity integrity". In this case, the AUTACK message must be secured (signed) by use of at least one other header and its corresponding trailer. It is advised that the header applicable for AUTACK security provides for encompassing security, encompassing the header used to indicate the referenced entity hashing method.

1.3.1.2 Independent signature of references.

The secured entity (message or interchange) is referenced as in the case above. Underneath it, at least one occurrence of USY is required referring to a security header of the present AUTACK, where a complete security (signature) method is described. This method is applied to the referred entity from its beginning service segment upto and including its ending service segment end. In the applicable header, the data-element "501 Security function, coded" refers to "reference" dentity NRO" (or to "referenced entity authentication"). In this case, the AUTACK message needs not be secured once more on itself. In case AUTACK is secured, it is advised that the " 0541 Scope of security application, coded" data-element of this AUTACK header, provides for encompassing security, encompassing the header occurrence(s) providing referenced entity security.

1.3.2 Use for acknowledgement and denial of acknowledgement.

Both cases are split at the AUTACK level. A same AUTACK should only acknowledge or deny acknowledgement. In either case, the AUTACK must be secured. Here too, at least one USY is required underneath the USX segment determining the acknowledged entity.

In case of non-acknowledgement, an error-code value must be provided to the data-element 0571. The default value of the error-code is 0, meaning "no security error". The "0534 security result link" applies now to the security method found in the original message, under the same number. The possible validation result present in the USY, is a repetition of the value found under the same number in the original message (or interchange).

Now at least one security method should be applied to AUTACK, to authenticate the (non-)acknowledgement.

2. References

See UNTDID, Part 4, Section 2.5, UN/ECE UNSM General Introduction, Section 1.

3. Terms and Definitions

See UNTDID, Part 4, Section 2.5, UN/ECE UNSM General Introduction, Section 2.

4. Message Definition

4.1 Data Segment Clarification

This section should be read in conjunction with the Branching Diagram and Segment Table which indicate mandatory, conditional and repeating requirements.

UNH, Message header A service segment starting and uniquely identifying a message. the message type identifier for the UN Secure Authentication and Acknowledgement Message is AUTACK.

Note. AUTACK message conforming to this issue 1 of the UNTDID must contain the following data in UNH, composite S009:

Data element	0065: AUTACK
	0052: 1
	0054: 1
	0051: UN

Segment Group 1: USH-USA-SG2
A group of segments providing all security information necessary for the integrity, authentication, and non-repudiation of origin or of receipt of all entities referenced in the AUTACK message and of the AUTACK message itself.

USH, Security header
A segment defining the security function being provided, the required response, the encoding rules on cryptographical data, the role of the security provider, the identification details of the originator and recipient and reference and time stamp data.

USA, Security algorithm
A segment used to identify a symmetric algorithm or hash function applied to the AUTACK, or to entities referred in the AUTACK.

Segment Group 2: USC-USA-USR
A group of segments providing the public key and its validation by a known or accepted authority.

USC, Certificate
A segment containing global parameters about the certificate layout, identification of the issuer and of the owner, the time stamp and validity period: start validity date and end validity date.

USA, Security algorithm
A segment to identify an algorithm used in the certificate.
Following algorithm identifications are required, in case of a full certificate: one for the hash function of the certification authority, one for the signature algorithm of this authority and one for the algorithm and public key value of the owner of the certificate. This last algorithm, when linked to the owner's identity define the purpose of the certificate.
Note: the public key value of the certification authority's signature algorithm should be obtained independently.

USR, Security result
A segment containing the signature by the certification authority on the remainder of the certificate.

USB, Beginning of a security message
A segment to describe the AUTACK function: authentication or acknowledgement, the wanted response, the message time stamp, related information and identification of the interchange sender and recipient of the secured entities.

Segment Group 3: USX-USY
A group of segments identifying the messages or interchanges being secured by the present AUTACK and containing their integrity or validation result.

USX, Security references
A segment referring to the secured entity and its creation date and time.

USY, Security on references
A segment used to identify the applicable header, contain the security result and indicate a possible cause of security rejection for the refered security link value.

Segment Group 4: UST-USR
A group of segments containing the results of the security method applied to the present AUTACK.

UST, Security trailer
A segment providing a link to the applicable security method header.

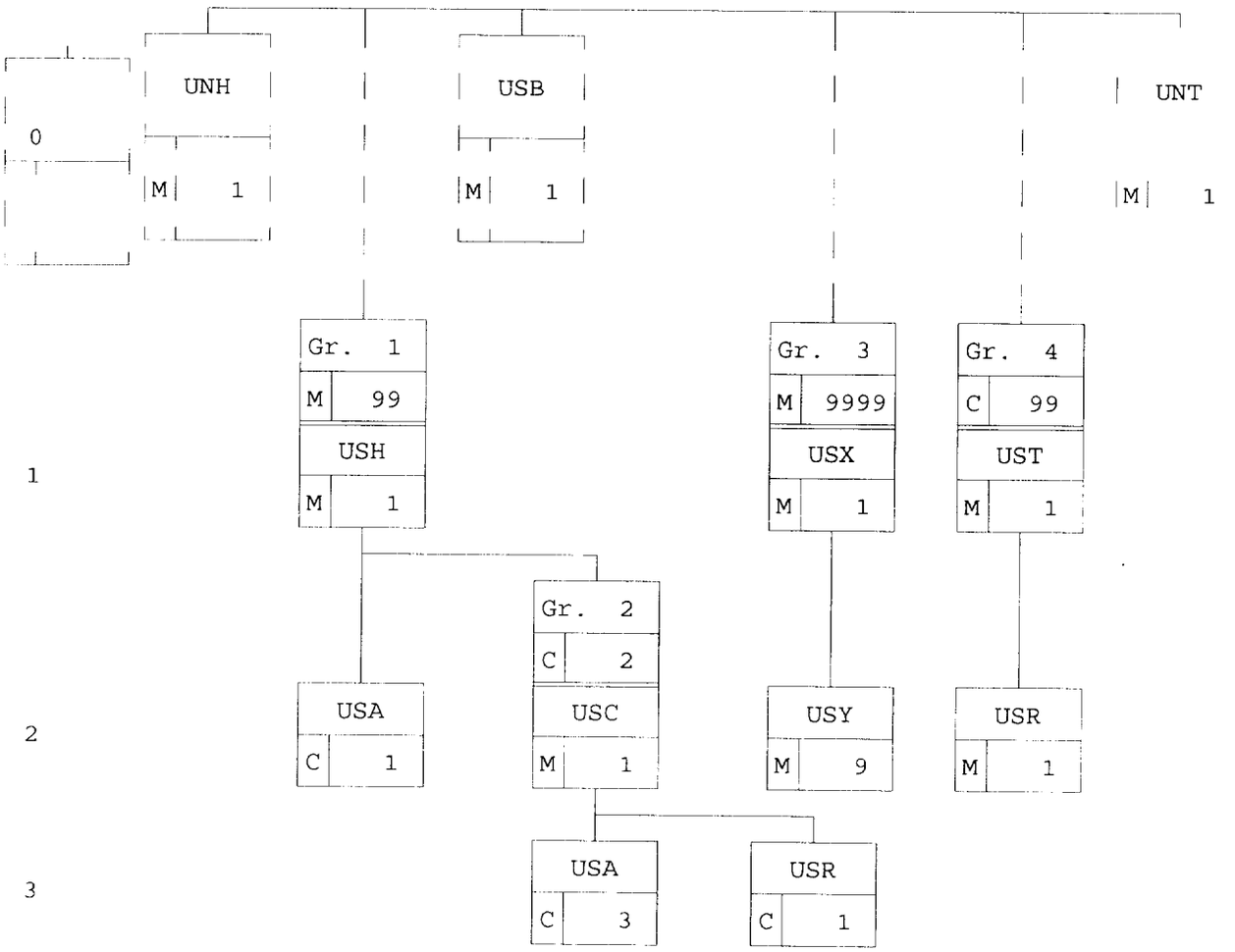
USR, Security result
A segment containing the security result of the applied security mechanism to the present AUTACK.

UNT, Message trailer
A service segment ending a message, giving the total number of segments in the message and the message reference number.

4.2 Message Structure

4.2.1 Branching Diagram

Level



4.2.2 Segment Table

TAG	NAME	S	REPT
UNH	Message header	M	1
	Segment Group 1	M	99
USH	Security header	M	1
USA	Security algorithm	C	1
	Segment Group 2	C	2
USC	Certificate	M	1
USA	Security algorithm	C	3
USR	Security result	C	1
USB	Beginning of a security message	M	1
	Segment Group 3	M	9999
USX	Security references	M	1
USY	Security on references	M	9
	Segment Group 4	C	99
UST	Security trailer	M	1
USR	Security result	M	1
UNT	Security trailer	M	1

4.3 Data Segment Index (Alphabetic Sequence)

UNH Message header
UNT Message trailer
USA Security algorithm
USB Beginning of a security message
USC Certificate
USH Security header
USR Security result
UST Security trailer
USX Security references
USY Security on references

5. Code values

To find code values one should consult the EDIFACT Security Implementation Guidelines. However, AUTACK needs a few values more.

5.1 Code values of the "0501 Security function, coded".

In addition to the code values foreseen for data element 0501 in the Security Implementation Guidelines, following values apply to AUTACK, skipping the value 4 reserved for confidentiality (encryption):

Code	Mnemonic	Meaning	Description
1	NRO	Non-repudiation of origin	See SIG
2	AUT	Message origin authentication	See SIG
3	INT	Integrity	See SIG
5	NRR	Non-repudiation of receipt	Service delivered by the recipient making denial of reception impossible
6	AUR	Receipt Authentic.	Service delivered by the recipient assuring secure reception.
7	RNO	Referenced entity NRO	Service assuring non-repudiation of origin on the referenced entity.
8	RAU	Referenced entity AUT	Service assuring authentication on referenced entity.
9	RINT	Referenced entity INT	Service assuring integrity on the referenced entity.

5.2 Code values for "0517 Date and time qualifier, coded".

Code	Mnemonic	Meaning	Description
1	STS	Security time stamp	See SIG
2	CGT	Certificate generation time	See SIG
3	CSV	Certificate start validity	See SIG
4	CEV	Certificate end validity	See SIG
5	MGT	Message generation d/t	Date and time at which the secured entity was generated.

5.3 Code values for "0563 Message function, coded".

Code	Mnemonic	Meaning	Description
1	AUT	Authentication	The message is used for authentication and / or non-repudiation of origin.
2	ACK	Acknowledgement	The message is used to acknowledge receipt of messages and interchanges.
3	NA	Non-acknowledge	The message is used to refuse acknowledgement, and mention security errors.

5.4 Code values for "0571 Security error, coded".

Code	Mnemonic	Meaning	Description
0	NULL	no error	
1	NAUT	wrong authenticator	The validation value is wrong.
2	NCER	wrong certificate	The certificate is wrong.
3	NPATH	certification path	The certification path is incomplete. Cannot verify.
4	NALG	not supported	The algorithm is not supported.
5	NHASH	not supported	Hashing method is not supported.
999	NDOC	not documented	

S002	INTERCHANGE SENDER			
Desc:	Identification of the sender of the interchange.			
Cont:	0004	Sender identification	M	an..35
	0007	Partner identification code qualifier	C	an..4
	0008	Address for reverse routing	C	an..14
S003	INTERCHANGE RECIPIENT			
Desc:	Identification of the recipient of the interchange.			
Cont:	0010	Recipient identification	M	an..35
	0007	Partner identification code qualifier	C	an..4
	0014	Routing address	C	an..14
S009	MESSAGE IDENTIFIER			
Desc:	Identification of the type, version etc. of the message being interchanged.			
Cont:	0065	Message type identifier	M	an..6
	0052	Message type version number	M	an..3
	0054	Message type release number	M	an..3
	0051	Controlling agency	M	an..2
	0057	Association assigned code	C	an..6
S010	STATUS OF THE TRANSFER			
Desc:	Statement that the message is one in a sequence of transfers relating to the same topic.			
Cont:	0070	Sequence message transfer number	M	n..2
	0073	First/last sequence message transfer indication	C	a1

```
+ S500  SECURITY IDENTIFICATION DETAILS                          USH, USC
  Desc: Identification of a security party.
  Cont: 0577  Security party qualifier                            M  an..3
        0538  Key name                                            C  an..35
        0511  Party id identification                             C  an..17
        0513  Code list qualifier                                 C  an..3
        0515  Code list responsible agency, coded                 C  an..3
        0586  Party name                                          C  an..35
        0586  Party name                                          C  an..35
        0586  Party name                                          C  an..35
-------------------------------------------------------------------------------
+ S501  SECURITY DATE AND TIME                                   USH USC USB USX
  Desc: Specification of a date and time for security purposes.
  Cont: 0517  Date and time qualifier, coded                      M  an..3
        0502  Date                                                C  n8
        0504  Time                                                C  n6
        0506  UTC offset                                          C  an..5
-------------------------------------------------------------------------------
+ S502  SECURITY ALGORITHM                                       USA
  Desc: Identifies the algorithm and its usage.
  Cont: 0523  Use of algorithm, coded                             M  an..3
        0525  Cryptographic mode of operation, coded              C  an..3
        0533  Mode of operation code list identifier              C  an..3
        0527  Algorithm, coded                                    C  an..3
        0529  Algorithm code list identifier                      C  an..3
-------------------------------------------------------------------------------
+ S503  ALGORITHM PARAMETER                                      USA
  Desc: Specification of a parameter needed for the algorithm.
  Cont: 0532  Algorithm parameter value                           C  an..512
        0531  Algorithm parameter qualifier                       C  an..3
-------------------------------------------------------------------------------
+ S505  SEPARATOR FOR SIGNATURE                                  USC
  Desc: Identifies a symbol used as syntactical separator.
  Cont: 0548  Separator for signature                             M  an..4
        0551  Separator for signature qualifier                   C  an..3
-------------------------------------------------------------------------------
+ S508  VALIDATION RESULT                                        USR
  Desc: Result of the applied algorithm.
  Cont: 0560  Validation value                                    M  an..
        0560  Validation value                                    C  an..
-------------------------------------------------------------------------------


  0004  Sender identification
  Desc: Name or coded representation of the sender of a data
        interchange.
  Repr: an..35                    Min: 1      Max: 35     Datatype: an
-------------------------------------------------------------------------------
  0007  Partner identification code qualifier
  Desc: Qualifier referring to the source of codes for the identifiers
        of interchanging partners.
  Repr: an..4                     Min: 1      Max: 4      Datatype: id
-------------------------------------------------------------------------------
```

0008 Address for reverse routing
Desc: Address specified by the sender of an interchange to be included by the recipient in the response interchanges to facilitate internal routing.
Repr: an..14 Min: 1 Max: 14 Datatype: an

0010 Recipient identification
Desc: Name or coded representation of the recipient of a data interchange.
Repr: an..35 Min: 1 Max: 35 Datatype: an

0014 Routing address
Desc: Address specified by the recipient of an interchange to be included by the sender and used by the recipient for routing of received interchanges inside his organization.
Repr: an..14 Min: 1 Max: 14 Datatype: an

0020 Interchange control reference
Desc: Unique reference assigned by the sender to an interchange.
Repr: an..14 Min: 1 Max: 14 Datatype: an

0051 Controlling agency
Desc: Code identifying the agency controlling the specification, maintenance and publication of the message type.
Repr: an..2 Min: 1 Max: 2 Datatype: id

0052 Message type version number
Desc: Version number of a message type.
Repr: an..3 Min: 1 Max: 3 Datatype: an

0054 Message type release number
Desc: Release number within the current message type version number (0052).
Repr: an..3 Min: 1 Max: 3 Datatype: an

0057 Association assigned code
Desc: Code, assigned by the association responsible for the design and maintenance of the message type concerned, which further identifies the message.
Repr: an..6 Min: 1 Max: 6 Datatype: id

0062 Message reference number
Desc: Unique message reference assigned by the sender.
Repr: an..14 Min: 1 Max: 14 Datatype: an

0065 Message type identifier
Desc: Code identifying a type of message and assigned by its controlling agency.
Repr: an..6 Min: 1 Max: 6 Datatype: id

0068 Common access reference
Desc: Reference serving as a key to relate all subsequent transfers of data to the same business case or file.
Repr: an..35 Min: 1 Max: 35 Datatype: an

0070 Sequence message transfer number
Desc: Number assigned by the sender indicating that the message is an addition or change of a previously sent message relating to

the same topic.
Repr: n..2 Min: 1 Max: 2 Datatype: n

0073 First/last sequence message transfer indication
Desc: Indication used for the first and last message in a sequence of the same type of message relating to the same topic.
Repr: a1 Min: 1 Max: 1 Datatype: id

0074 Number of segments in a message
Desc: Control count of number of segments in a message
Repr: n..6 Min: 1 Max: 6 Datatype: n

+ 0501 Security function, coded USH
Desc: Specification of the security function applied in the message or interchange.
Repr: an..3 Min: 1 Max: 3 Datatype: an

+ 0502 Date S501
Desc: Indication of a date in YYYYMMDD (century included)
Repr: n8 Min: 8 Max: 8 Datatype: n

+ 0503 Response type, coded USH
Desc: Code specifying the type of acknowledgment required.
Repr: an..3 Min: 1 Max: 3 Datatype: an

+ 0504 Time S501
Desc: Indication of a time in HHMMSS.
Repr: n6 Min: 6 Max: 6 Datatype: n

+ 0505 Filter function, coded. USH - USC
Desc: Identification of the filtering function used for validation results, keys and ciphertext.
Repr: an..3 Min: 1 Max: 3 Datatype: an

+ 0506 UTC offset S501
Desc: Offset from the UTC standard time.
Repr: an..5 Min: 1 Max: 5 Datatype: an

+ 0507 Character set encoding, coded USH - USC
Desc: Identifies the characterset (binary code) in which the message was coded when security mechanisms were applied.
Repr: an..3 Min: 1 Max: 3 Datatype: an

+ 0509 Role of security provider, coded USH
Desc: Identifies the function of the security provider in relation to the secured item.
Repr: an..3 Min: 1 Max: 3 Datatype: an

+ 0511 Party id identification S500
Desc: Code identifying a party involved in a transaction.
Repr: an..17 Min: 1 Max: 17 Datatype: an

```
+ 0513  Code list qualifier                                    S500
  Desc: Identification of a code list.
  Repr: an..3                    Min: 1      Max: 3      Datatype: an
-------------------------------------------------------------------------
+ 0515  Code list responsible agency, coded                    S500
  Desc: Code identifying an agency responsible for a code list.
  Repr: an..3                    Min: 1      Max: 3      Datatype: an
-------------------------------------------------------------------------
+ 0516  Security reference number                              USH
  Desc: An element allowing identification for sequence integrity
        between a security originator and a security recipient.
  Repr: an..35                   Min: 1      Max: 35     Datatype: an
-------------------------------------------------------------------------
+ 0517  Date and time qualifier, coded                         S501
  Desc: Code giving specific meaning to a date and time.
  Repr: an..3                    Min: 1      Max: 3      Datatype: an
-------------------------------------------------------------------------
+ 0523  Use of algorithm, coded                               S502
  Desc: Specifies the usage made of the algorithm.
  Repr: an..3                    Min: 1      Max: 3      Datatype: an
-------------------------------------------------------------------------
+ 0525  Cryptographic mode of operation, coded                S502
  Desc: Identifies the cryptographic mode of operation.
  Repr: an..3                    Min: 1      Max: 3      Datatype: an
-------------------------------------------------------------------------
+ 0527  Algorithm, coded                                      S502
  Desc: Identifies the algorithm used.
  Repr: an..3                    Min: 1      Max: 3      Datatype: an
-------------------------------------------------------------------------
+ 0529  Algorithm code list identifier                        S502
  Desc: Identification of an algorithm code list.
  Repr: an..3                    Min: 1      Max: 3      Datatype: an
-------------------------------------------------------------------------
+ 0531  Algorithm parameter qualifier                         S503
  Desc: Specifies the used parameter.
  Repr: an..3                    Min: 1      Max: 3      Datatype: an
-------------------------------------------------------------------------
+ 0532  Algorithm parameter value                             S503
  Desc: Contains the value of the parameter, filtered if required and
        cleaned for possible separators by use of the release
        character.
  Repr: an..512                  Min: 1      Max: 512    Datatype: an
-------------------------------------------------------------------------
+ 0533  Mode of operation code list identifier                S502
  Desc: Specifies the code list used to identify the cryptographic
        mode of operation
  Repr: an..3                    Min: 1      Max: 3      Datatype: an
-------------------------------------------------------------------------
+ 0534  Security result link.                                 UST USH USY
  Desc: Contains a number which links a particular security header
        group with the corresponding security trailer group or with
        referred entity security result.
  Repr: n2                       Min: 2      Max: 2      Datatype: n
-------------------------------------------------------------------------
```

+ 0536	Certificate reference			USC
Desc:	Reference number assigned to the certificate by the issuer.			
Repr:	an..35	Min: 1	Max: 35	Datatype: an
+ 0538	Key name			S500
Desc:	Identification of a key.			
Repr:	an..35	Min: 1	Max: 35	Datatype: an
+ 0541	Scope of security application, coded			USH
Desc:	Specifies the scope of application of the security service.			
Repr:	an..3	Min: 1	Max: 3	Datatype: an
+ 0543	Character set repertoire, coded			USC
Desc:	Identifies the character set repertoire (UNOA to UNOF) used when the certificate was signed.			
Repr:	an..3	Min: 1	Max: 3	Datatype: an
+ 0544	Certificate format version			USC
Desc:	Version number of the format of the certificate, identified by year and status of the UN/EDIFACT service segment directory			
Repr:	an..3	Min: 1	Max: 3	Datatype: an
+ 0546	User authorisation levels			USC
Desc:	Specification of the attributes associated with the owner of the certificate.			
Repr:	an..35	Min: 1	Max: 35	Datatype: an
+ 0548	Separator for signature			S505
Desc:	Separator used when the signature was computed.			
Repr:	an..4	Min: 1	Max: 4	Datatype: an
+ 0551	Separator for signature qualifier			S505
Desc:	Identifies each UN/EDIFACT separator or release character.			
Repr:	an..3	Min: 1	Max: 3	Datatype: an
+ 0552	Security structure version number			USH
Desc:	Version number of the format of the security headers and trailers identified by year and status of of the UN/EDIFACT service segment directory.			
Repr:	an..3	Min: 1	Max: 3	Datatype: an
+ 0560	Validation value			S508
Desc:	Gives a result of the application of the security algorithm to the secured entity, filtered if required and cleaned from possible separators by use of the release character.			
Repr:	an..256	Min: 1	Max: 256	Datatype: an
+ 0563	Message function, coded			USB
Desc:	Code indicating the function of the message.			
Repr:	an..3	Min: 1	Max: 3	Datatype: an
+ 0571	Security error, coded			USY
Desc:	Identifies the security cause for the rejection of the entity.			
Repr:	an..3	Min: 1	Max: 3	Datatype: an

```
+ 0577  Security party qualifier                              S500
  Desc: Specification of the function of the security party
        identified.
  Repr: an..3                    Min: 1      Max: 3      Datatype: an
-------------------------------------------------------------------------
+ 0586  Party name                                            S500
  Desc: Name of a party involved in a transaction.
  Repr: an..35                   Min: 1      Max: 35     Datatype: an
-------------------------------------------------------------------------
+ 0590  Related filename                                      USB
  Desc: Gives the related filename used for compatibility with BCS
        (Banking Communication Standard)
  Repr: an..256                  Min: 1      Max: 256    Datatype: an
-------------------------------------------------------------------------
```

TAG	NAME	S	REPT
UNH	Message header	M	1
----	Segment Group 1	M	99
USH	Security header	M	1
USA	Security algorithm	C	1
----	Segment Group 2	C	2
USC	Certificate	M	1
USA	Security algorithm	C	3
USR	Security result	C	1
USB	Beginning of a security message	M	1
----	Segment Group 3	M	9999
USX	Security references	M	1
USY	Security on references	M	9
----	Segment Group 4	C	99
UST	Security trailer	M	1
USR	Security result	M	1
UNT	Message trailer	M	1

UNH MESSAGE HEADER

Function: To head, identify and specify a message.

0062	MESSAGE REFERENCE NUMBER	M	an..14
S009	MESSAGE IDENTIFIER	M	
0065	Message type identifier	M	an..6
0052	Message type version number	M	an..3
0054	Message type release number	M	an..3
0051	Controlling agency	M	an..2
0057	Association assigned code	C	an..6
0068	COMMON ACCESS REFERENCE	C	an..35

S010	STATUS OF THE TRANSFER	C	
0070	Sequence message transfer number	M	n..2
0073	First/last sequence message transfer indicati	C	a1

UNT MESSAGE TRAILER

Function: To end and check the completeness of a message.

0074	NUMBER OF SEGMENTS IN A MESSAGE	M	n..6
0062	MESSAGE REFERENCE NUMBER	M	an..14

+ USA SECURITY ALGORITHM AUTACK

Function: To identify an algorithm, its technical usage and its associated technical parameters.

S502	SECURITY ALGORITHM	M	
0523	Use of algorithm, coded	M	an..3
0525	Cryptographic mode of operation, coded	C	an..3
0533	Mode of operation code list identifier	C	an..3
0527	Algorithm, coded	C	an..3
0529	Algorithm code list identifier	C	an..3
S503	ALGORITHM PARAMETER	C	
0532	Algorithm parameter value	C	an..512
0531	Algorithm parameter qualifier	C	an..3
S503	ALGORITHM PARAMETER	C	
0532	Algorithm parameter value	C	an..512
0531	Algorithm parameter qualifier	C	an..3
S503	ALGORITHM PARAMETER	C	
0532	Algorithm parameter value	C	an..512
0531	Algorithm parameter qualifier	C	an..3
S503	ALGORITHM PARAMETER	C	
0532	Algorithm parameter value	C	an..512
0531	Algorithm parameter qualifier	C	an..3
S503	ALGORITHM PARAMETER	C	
0532	Algorithm parameter value	C	an..512
0531	Algorithm parameter qualifier	C	an..3

+ USB BEGINNING OF A SECURITY MESSAGE AUTACK

Function: To indicate the type, function and related details of a security message.

0563	MESSAGE FUNCTION, CODED	C	an..3
0503	RESPONSE TYPE, CODED	C	an..3
S501	SECURITY DATE AND TIME	C	
0517	Date and time qualifier, coded	M	an..3
0502	Date	C	n8
0504	Time	C	n6
0506	UTC offset	C	an..5
0590	RELATED FILENAME	C	an..256
S002	INTERCHANGE SENDER	C	
0004	Sender identification	M	an..35
0007	Partner identification code qualifier	C	an..4
0008	Address for reverse routing	C	an..14
S003	INTERCHANGE RECIPIENT	C	
0010	Recipient identification	M	an..35
0007	Partner identification code qualifier	C	an..4
0014	Routing address	C	an..14

+ USC CERTIFICATE AUTACK

Function: To contain the data necessary to validate the asymmetric security methods applied to a message or an interchange.

0536	CERTIFICATE REFERENCE	C	an..35
S500	SECURITY IDENTIFICATION DETAILS	C	
0577	Security party qualifier	M	an..3
0538	Key name	C	an..35
0511	Party id identification	C	an..17
0513	Code list qualifier	C	an..3
0515	Code list responsible agency, coded	C	an..3
0586	Party name	C	an..35
0586	Party name	C	an..35
0586	Party name	C	an..35

S500	SECURITY IDENTIFICATION DETAILS	C	
0577	Security party qualifier	M	an..3
0538	Key name	C	an..35
0511	Party id identification	C	an..17
0513	Code list qualifier	C	an..3
0515	Code list responsible agency, coded	C	an..3
0586	Party name	C	an..35
0586	Party name	C	an..35
0586	Party name	C	an..35
0544	CERTIFICATE FORMAT VERSION	C	an..3
0505	FILTER FUNCTION, CODED.	C	an..3
0507	CHARACTER SET ENCODING, CODED	C	an..3
0543	CHARACTER SET REPERTOIRE, CODED	C	an..3
0546	USER AUTHORISATION LEVELS	C	an..35
S505	SEPARATOR FOR SIGNATURE	C	
0548	Separator for signature	M	an..4
0551	Separator for signature qualifier	C	an..3
S505	SEPARATOR FOR SIGNATURE	C	
0548	Separator for signature	M	an..4
0551	Separator for signature qualifier	C	an..3
S505	SEPARATOR FOR SIGNATURE	C	
0548	Separator for signature	M	an..4
0551	Separator for signature qualifier	C	an..3
S505	SEPARATOR FOR SIGNATURE	C	
0548	Separator for signature	M	an..4
0551	Separator for signature qualifier	C	an..3
S501	SECURITY DATE AND TIME	C	
0517	Date and time qualifier, coded	M	an..3
0502	Date	C	n8
0504	Time	C	n6
0506	UTC offset	C	an..5
S501	SECURITY DATE AND TIME	C	
0517	Date and time qualifier, coded	M	an..3
0502	Date	C	n8
0504	Time	C	n6
0506	UTC offset	C	an..5
S501	SECURITY DATE AND TIME	C	
0517	Date and time qualifier, coded	M	an..3
0502	Date	C	n8
0504	Time	C	n6
0506	UTC offset	C	an..5

+ USH SECURITY HEADER AUTACK

Function: To specify the security service applied.

0552	SECURITY STRUCTURE VERSION NUMBER	M	an..3
0501	SECURITY FUNCTION, CODED	M	an..3
0534	SECURITY RESULT LINK.	M	n2
0541	SCOPE OF SECURITY APPLICATION, CODED	C	an..3
0503	RESPONSE TYPE, CODED	C	an..3
0505	FILTER FUNCTION, CODED.	C	an..3
0507	CHARACTER SET ENCODING, CODED	C	an..3
0509	ROLE OF SECURITY PROVIDER, CODED	C	an..3
S500	SECURITY IDENTIFICATION DETAILS	C	
0577	Security party qualifier	M	an..3
0538	Key name	C	an..35
0511	Party id identification	C	an..17
0513	Code list qualifier	C	an..3
0515	Code list responsible agency, coded	C	an..3
0586	Party name	C	an..35
0586	Party name	C	an..35
0586	Party name	C	an..35
S500	SECURITY IDENTIFICATION DETAILS	C	
0577	Security party qualifier	M	an..3
0538	Key name	C	an..35
0511	Party id identification	C	an..17
0513	Code list qualifier	C	an..3
0515	Code list responsible agency, coded	C	an..3
0586	Party name	C	an..35
0586	Party name	C	an..35
0586	Party name	C	an..35
0516	SECURITY REFERENCE NUMBER	C	an..35
S501	SECURITY DATE AND TIME	C	
0517	Date and time qualifier, coded	M	an..3
0502	Date	C	n8
0504	Time	C	n6
0506	UTC offset	C	an..5

+ USR SECURITY RESULT AUTACK

Function: To contain the security result corresponding to the security service specified in the applicable USA segment.

S508	VALIDATION RESULT	M	
0560	Validation value	M	an..256
0560	Validation value	C	an..256

+ UST SECURITY TRAILER AUTACK

Function: To link to the security header.

0534	SECURITY RESULT LINK.	M	n2

+ USX SECURITY REFERENCES AUTACK

Function: To identify a message or an interchange in the security process .

0020	INTERCHANGE CONTROL REFERENCE	M	an..14
0062	MESSAGE REFERENCE NUMBER	C	an..14
S501	SECURITY DATE AND TIME	C	
0517	Date and time qualifier, coded	M	an..3
0502	Date	C	n8
0504	Time	C	n6
0506	UTC offset	C	an..5

+ USY SECURITY ON REFERENCES AUTACK

Function: To give security information on the refered message or interchange.

0534	SECURITY RESULT LINK.	M	n2
S508	VALIDATION RESULT	C	
0560	Validation value	M	an..256
0560	Validation value	C	an..256
0571	SECURITY ERROR, CODED	C	an..3

EXHIBIT B-4

UNITED NATIONS

E

Economic and Social Council

Distr.

RESTRICTED

TRADE/WP.4/R.1026/Add.4
22 February 1994

ENGLISH ONLY

ECONOMIC COMMISSION FOR EUROPE

COMMITTEE ON THE DEVELOPMENT OF TRADE

Working Party on Facilitation of
International Trade Procedures
(Item 7 of the provisional agenda of
the Meetings of Experts on Data Elements
and Automatic Data Interchange (GE.1)
Forty-ninth session, 15-16 March 1994)

MIG HANDBOOK
UN/EDIFACT MESSAGE AUTACK

* * *

Submitted by the Rapporteur for the West European EDIFACT Board *

* The present document is reproduced in the form in which it was received by the secretariat.

GE.94- 30555

MIG HANDBOOK

UN/EDIFACT MESSAGE AUTACK

Message Type Message)	:	AUTACK (Secure Authentication and Acknowledgement
Release	:	1
Version	:	4
Status	:	1
Date	:	16-02-1994

Developed by the Western European EDIFACT Board Special Interest Group Security, in collaboration with PAEB and ANEB

Published by Rabobank Nederland

This Message Implementation Guidelines Handbook (MIG) describes the use of the Secure Authentication and Acknowledgement Message (AUTACK) which reached Status 1 in 1994. It is based on the Service Directory 1 approved by UN/ECE WP.4

TABLE OF CONTENTS

1 SCOPE

The AUTACK message provides security services of messages forwarded separately. It can be used for any combination of messages that need to be secured between two parties.

A secure authentication and acknowledgement message used as an authentication message is sent by the originator of messages and interchanges forwarded separately or by a party having authority to act on behalf of the originator, to facilitate message origin authentication, validation of integrity of content, validation of message sequence integrity or non-repudiation of origin of these messages.

A secure authentication and acknowledgement message used as an acknowledgement message is sent by the recipient of previously received messages or by a party having authority to act on behalf of the recipient, to facilitate confirmation of receipt, validation of integrity of content, validation of completeness or non-repudiation of receipt of these messages.

This AUTACK message can apply to one or more messages from one or more interchanges, or to one or more interchanges.

The security services are provided by cryptographic mechanisms applied to the content of the original messages or interchanges. The results of this mechanisms form the body of the AUTACK message, supplemented by the relevant data such as the reference to the cryptographic methods used, the reference number and the date and time of the original entities.

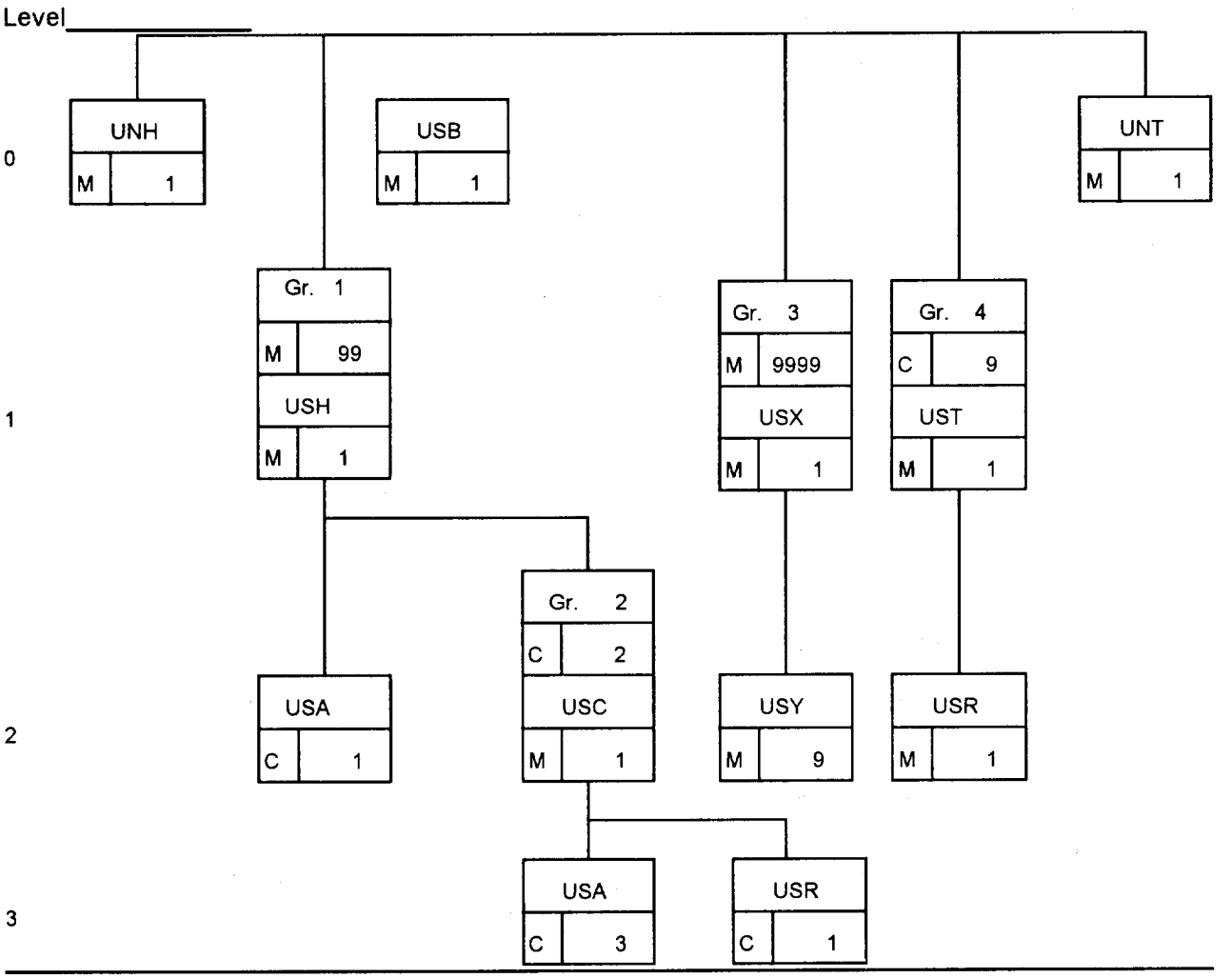
The AUTACK message is completed by the standard security header and trailer groups, which are mandatory for this message.

2 FORMAT SPECIFICATIONS

2.1 Segment Table

TAG	NAME	S	REPT
UNH	**Message header**	**M**	**1**
------ Segment Group 1 -----		M	99
USH	Security header	M	1
USA	Security algorithm.	C	1
----- Segment Group 2 -----		C	2
USC	Certificate	M	1
USA	Security algorithm.	C	3
USR	Security result.	C	1
USB	Beginning of a security message	M	1
----- Segment Group 3 -----		M	9999
USX	Security references	M	1
USY	Security on references	C	9
----- Segment Group 4 -----		C	9
UST	Security trailer	M	1
USR	Security result.	M	1
UNT	Message trailer	M	1

2.2 Branching diagram



3 DATA SEGMENT SPECIFICATIONS

3.1 UNH, MESSAGE HEADER (Mandatory, 1)

3.1.1 Segment Format

Number	Description	M/C	Format	Special Notes
0062	MESSAGE REFERENCE NUMBER	M	an..14	
S009	MESSAGE IDENTIFIER	M		
0065	Message type identifier	M	an..6	"AUTACK"
0052	Message type version number	M	an..3	"1"
0054	Message type release number	M	an..3	"1"
0051	Controlling agency	M	an..2	"UN"
0057	Association assigned code	C	an..6	
0068	COMMON ACCESS REFERENCE	C	an..35	
S010	STATUS OF THE TRANSFER	C		
0070	Sequence message transfer number	M	n..2	
0073	First/last sequence message transfer indication	C	al	

3.1.2 Segment Description

A service segment starting and uniquely identifying a message. This segment also specifies the message type version and release numbers used in formatting the information and identifies the agency responsible for maintenance of the message.

3.1.3 Segment Rules

3.1.3.1 MESSAGE REFERENCE NUMBER (0062)

Identification of a message by a unique number within a range of messages sent (e.g. the first message sent is 1, the second 2, etc.).

3.1.3.2 MESSAGE IDENTIFIER (S009)

Specification of the message type being sent (e.g. Payment Order), followed by the version number and release number used in formatting the information. Use of version/release numbers other than the latest implemented **must** be bilaterally agreed outside the message.

3.1.3.3 Message type identifier (0065)

Identification of the EDIFACT message type contained within the Message Header (UNH) and Message Trailer (UNT) segments.

3.1.3.4 Message type version number (0052)

Identification of the EDIFACT version number of the message contained within the Message Header (UNH) and Message Trailer (UNT) segments.

3.1.3.5 Message type release number (0054)

Identification of the EDIFACT release number of the message contained within the Message Header (UNH) and Message Trailer (UNT) segments.

3.1.3.6 Controlling agency (0051)

Specification of the agency responsible for maintenance of the message type which has been identified. The code "UN" (i.e. EDIFACT Messages) **must always** be used.

3.1.3.7 Association assigned code (0057)

Code, assigned by the association responsable for the design and maintenance of the message type concerned, which further identifies the message.

3.1.3.8 Common access reference (0068)

Reference serving as a key to relate all subsequent transfers of data to the same business case or file.

3.1.3.9 Status of the transfer (S010)

Statement that the message is one in sequence of transfers relating to the same topic.

3.1.3.10 Sequence message transfer number (0070)

Number assigned by the sender indicating that the message is an addition or change of a previously sent message relating to the same topic

3.1.3.11 First / last sequence message transfer indication (0073)

Indication used for the first and last message in a sequence of the same type of message relating to the same topic.

3.1.4 EXAMPLE

SEGMENT GROUP 1 (Mandatory, 99)

USH	Security header	M	1
USA	Security algorithm	C	1

Segment Group 2 (Conditional, 2)			
USC	Certificate	M	1
USA	Security Algorithm	C	3
USR	Security Result	C	1

This segment group provides all security information necessary for the integrity, the authentication, the non-repudiation of origin or secure acknowledgement of all the messages referenced in the AUTACK message.

Note: Segment Groups 1 and 2 are "nested". The trigger segment is Segment Group 1. Segment Group 1 may be present alone, or Segment Groups 1 and 2 may both be present.

3.2 USH - SECURITY HEADER (Mandatory, 1)

3.2.1 Segment Format

Number	Description	M/C	Format	Special notes
0552	SECURITY STRUCTURE VERSION NUMBER	M	an..3	
0501	SECURITY FUNCTION CODED	M	an..3	
0534	SECURITY RESULT LINK	M	n2	
0541	SCOPE OF SECURITY APPLICATION CODED	C	an..3	
0503	RESPONSE TYPE, CODED	C	an..3	
0505	FILTER FUNCTION, CODED	C	an..3	
0507	CHARACTER SET ENCODING, CODED	C	an..3	
0509	ROLE OF SECURITY PROVIDER, CODED	C	an..3	
S500	SECURITY IDENTIFICATION DETAILS	C		
0577	Security party qualifier	M	an..3	
0538	Key name	C	an..35	
0511	Party Id identifaction	C	an..17	
0513	Code list qualifier	C	an.. 3	
0515	Code list responsible agency	C	an...3	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
S500	SECURITY IDENTIFICATION DETAILS	C		
0577	Security party qualifier	M	an...3	
0538	Key name	C	an..35	
0511	Party Identification	C	an..17	
0513	Code list qualifier	C	an...3	
0515	Code list responsible agency	C	an...3	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
0516	SECURITY REFERENCE NUMBER	C	an..35	
S501	SECURITY DATE AND TIME	C		
0517	Date and time qualifier, coded	M	an..3	
0502	Date	C	n8	
0504	Time	C	n6	
0506	UTC offset	C	an...5	

3.1.2 Segment Description

The SECURITY HEADER segment specifies the security service applied to the AUTACK message or the referenced message or interchange.

At least SECURITY STRUCTURE VERSION NUMBER (0552), SECURITY FUNCTION (0501) AND SECURITY RESULT LINK (0534) must be present.

The SECURITY IDENTIFICATION DETAILS composite segments in group 1 either:

- must be used if symmetric methods are used (see 3.2.2), or
- may be used if asymmetric methods are used, and if there is a need to distinguish between the
  security orginator certificate and the security recipent certificate.

3.2.3 Segment Rules

There may be several different USH segments within the same AUTACK, if different security functions are applied to the AUTACK message or the referenced entities (e. g. integrity and non-repudiation of origin) or if the same security function is simultaneously applied by several entities.

The SECURITY RESULT LINK will be used to establish a link between one USH segment and the related USR or USY segment, depending on the security function.

3.2.3.1 SECURITY STRUCTURE VERSION NUMBER (0552)

It specifies the version number of the format of the security headers and trailers identified by year and status of the UN/EDIFACT service segment directory.

3.2.3.2 SECURITY FUNCTION, CODED (0501)

It specifies the security function applied to the message. One of the following codes **must** be used:

Code	Mnemo.	Meaning	Description
1	NRO	Non-repudiation of origin	The message includes a digital signature protecting the receiver of the message from the sender's denial of having sent the message.
2	AUT	Message origin authentication	The actual sender of the message cannot claim to be some other (authorized) entity.
3	INT	Integrity	The message content is protected against the modification of data
5	NRR	Non-repudiation of receipt	Service delivered by the recipient making denial of reception impossible
6	AUR	Receipt Authentic.	Service delivered by the recipient assuring secure reception
7	RNO	Referenced entit. NRO	Service assuring non-repudiation of origin on the referenced entity
8	RAU	Referenced entit. AUT	Service assuring authentication on referenced entity
9	RINT	Referenced entit.	Service assuring integrity on the referenced entity

The security functions are described in the document "Recommendations for UN/EDIFACT message level security from the UN/EDIFACT Security Joint Working Group".
The security functions NRO, AUT and INT are applied to the AUTACK itself.
All the other security functions are connected to the referenced entities.
NRR and AUR are security functions of the AUTACK message used by the recipient and RNO, RAU and RINT by the sender.

3.2.3.3 SECURITY RESULT LINK (0534)

Contains a number which links a particular USH segment with its corresponding USR or USY segment. The value used is arbitrarily assigned but, within one message, the same value must not be used more than once.

The rules are:
Using NRO or AUT the link into: USR (group 4).
Using INT, NRR, AUR, RNO, RAU or RINT the link is to: USY (group 3).

Remark: Only in the case that digital signatures or message authentication codes of referenced entities are transported in the AUTACK message there is no link to the USR segment in group 4.

3.2.3.4 SCOPE OF SECURITY APPLICATION, CODED (0541)

It specifies the scope of application of the security service defined in the present header, thus it defines the data that have to be taken into account by the related cryptographic process.

This scope may be either :

- the **current Security Header segment group** (from the first character, namely a "U", to the separator ending this Security Header segment group, both included), and the **UNSM body** (from the first character following the separator ending the last Security Header segment group to the separator preceding the first character of the first Security Trailer segment group, both included). In this case any other Security Header or Security Trailer segment group will not be encompassed within this scope. The code used in this case is "SHM",

or

- from the first character (namely a "U", included) of the **current Security Header segment group** to the first character of the **related Security Trailer segment group** (namely a "U", included). The relation between the Security Header and Security Trailer segment groups is provided by the data elements security result link of the USH and of the UST segments. In this case, the scope of application of the security mechanism of the current header will encompass the UNSM body and all the Security Headers and Security Trailers embedded within this Security Header and its related Security Trailer. The code to use in this case is "SHT",

or

- from the first character of the message or interchange to the last character of the message or interchange.

One or the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	SHM	Security Header and Message body	see explanation above
2	SHT	From Security Header to Security Trailer	see explanation above
3	TOT	Whole related message or interchange	see explanation above

3.2.3.5 RESPONSE TYPE, CODED (0503)

It specifies whether a secure acknowledgment from the message recipient is required or not. If it is required, the message sender will expect an AUTACK message to be sent back by the current message recipient to the current message sender, containing this acknowledgement.

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	NA	No acknowledgement required	No AUTACK acknowledgment message expected
2	AC	Acknowledgement required	AUTACK acknowledgment message expected

3.2.3.6 FILTER FUNCTION, CODED (0505)

Identification of the filtering function used for validation results and keys .

One of the following codes **must** be used :

Code	Mnemo .	Meaning	Description
1	NUL	No filter	Self explanatory
2	HEX	Hexadecimal filter	Hexadecimal filter
3	ASC	ISO 646 filter	ASCII filter as described in DIS 10126-1 (see note)
4	BAU	ISO 646 Baudot filter	Baudot filter as described in DIS 10126-1
5	EDA	UN/EDIFACT level A filter	UN/EDIFACT level A filter function as described in Annex A of the present document
999	ZZZ	Mutually agreed	Self explanatory

note :

This filtering function is mentioned here for completeness, but does not comply with requirements of UN/EDIFACT level A or B syntax.

3.2.3.7 CHARACTER SET ENCODING, CODED (0507)

Identifies the character set in which the message was coded when security mechanisms were applied.

One of the following codes **must** be used :

Code	Mnemo .	Meaning	Description
1	AS7	ASCII 7 bit	Self explanatory
2	AS8	ASCII 8 bit	Self explanatory
3	EBC	EBCDIC IBM 360	IBM 360 machine EBCDIC code
4	IPC	ASCII IBM PC	IBM PC ASCII code
5	VAX	ASCII DEC VAX	DEC VAX ASCII 8 bit code
999	ZZZ	Mutually agreed	Self explanatory

3.2.3.8 ROLE OF SECURITY PROVIDER, CODED (0509)

Identifies the function of the security provider as to the secured item.

One of the following codes **must** be used :

Code	Mnemo	Meaning	Description
1	ISS	Issuer	The security provider is the rightful issuer of the signed document
2	NOT	Notary	The security provider acts as a notary in relation to the signed document
3	CON	Contracting party	The security provider endorses the content of the signed document
4	WIT	Witness	The security provider is a witness, but is not responsible for the content of the signed document
999	ZZZ	Undefined	The role of the security provider is not defined

note :

when this data element is not used, the value "ISS" is assumed.

3.2.3.9 SECURITY IDENTIFICATION DETAILS (S500)

Identification of parties involved in the security process. Two occurrences of this composite data element are possible : one for the security originator, one for the security recipient.

If asymmetric algorithms are used, party identification is performed by the use of certificates. Hence, this composite should be used either :

- if symmetric algorithms are used, or
- if asymmetric algorithms are used and when two certificates are present, in order to distinguish between the originator and the recipient certificates

If these composite data elements are used at least the Security party qualifier (identifying security originator or security recipient) **must** be present.

SECURITY IDENTIFICATION DETAILS composite data elements should be used only if the parties involved in security are not un-ambiguously identified by certificates (use of symmetric methods or asymmetric methods, to clarify the use of more than one certificate).

The identification of the SECURITY IDENTIFICATION DETAILS composite data elements related to the security originator and of the one related to the security recipient is achieved by the Security party qualifier.

If asymmetric methods are used, this SECURITY IDENTIFICATION DETAILS composite data element may be used to identify that a certificate is used by an entity acting as security originator or security recipient. In this case, the identification of the party of the SECURITY IDENTIFICATION DETAILS composite data element (any of the data elements 0511, 0513, 0515, 0586) in the USH segment group will be the same as the identification of the party, qualified as "certificate owner" in the USC segment group, and the security party qualifier will identify the function (originator or recipient) of the party involved.

Security party qualifier (0577)

Specification of the function of the security party identified. One of the following codes **must** be used :

Code	Mnemo	Meaning	Description
1	MS	Message sender	Identifies the party which generates the security parameters of the message (i.e. security originator).
2	MR	Message receiver	Identifies the party which verifies the security parameters of the message (i.e. security recipient).

Key name (0538)

Identification of a key.

The USA segment allows all the information related to the cryptographic mechanism applied to the secured message to be conveyed.This includes the identification of the key, in the case of symmetric algorithms.

If there is no need to convey a USA segment in the secured message (because the cryptographic mechanisms have been agreed previously between the partners), the data element key name (0538) of the USH segment may be used to establish the key relationship between the sending and receiving parties.

Nevertheless, it is stongly reommended to use either the data element key name (0538) in the USH segment, or the data element algorithm parameter value (0532) with the appropriate qualifier in the USA (data element algorithm parameter qualifier, with the code value "KYN"), but not both of them, within the same message Security Header.

Party Id identification (0511)

Code identifying a party involved in the security process, according to a defined registry of security parties.

Code list qualifier (0513)

Code identifying the type of identification used to register the security parties.

Code list responsible agency (0515)

Code identifying the agency in charge of registration of the security parties.

Party name (0586)

Identification of the security party. Three occurences may be used to allow a complete identification.

3.2.3.10 SECURITY REFERENCE NUMBER (0516)

Identification of the message to which security is applied. This identification is security related and may differ from the identification of the message that may appear elsewhere. The reference number may be used to provide message sequence integrity.

3.2.3.11 SECURITY DATE AND TIME (S501)

Security timestamp of the message to which security is applied. This timestamp is security related and may differ from any dates and times that may appear elsewhere in the message. It may be used to provide message sequence integrity.

Date and time qualifier (0517)

Specification of the type of date and time. The following code **must** be used :

Code	Mnemo.	Meaning	Description
1	STS	Security Timestamp	Security timestamp of the secured messsage

Date (0502)

Specification of the date. Its format **must** be YYYYMMDD (century included).

Time (0504)

Specification of the time. Its format **must** be HHMMSS (HH being in 24 hour clock format).

UTC offset (0506)

Offset from UTC standard time. Its format may be either :

XHHMM (X being P for plus or M for minus, followed by the offset, in hours and minutes), or

XY (X being : C = Central, E = Eastern, M = Mountain, P = Pacific, and Y being : D = daylight time, S = Standard time, T = Time).

3.2.4. EXAMPLE

3.3 USA - SECURITY ALGORITHM

3.3.1 Segment Format

Number	Description	M/C	Format	Special notes
S502	SECURITY ALGORITHM	M		
0523	Use of algorithm, coded	M	an..3	
0525	Cryptographic mode of operation, coded	C	an..3	
0533	Mode of operation code list identifier	C	an..3	
0527	Algorithm, coded	C	an..3	
0529	Algorithm code list identifier	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier	C	an..3	

3.3.2 Segment Description

This segment is used to identify an algorithm, the technical usage made of it, and to contain the technical parameters required.

The use of algorithm data element specifies the usage made of the algorithm, the algorithm data element identifies the algorithm itself, and the cryptographic mode of operation specifies the mode of operation used.

The mode of operation code list identifier and the algorithm code list identifier identify in which code list the codes of the preceding data elements (mode of operation or algorithm) are defined.

The algorithm parameter composite data element provides space for one parameter. It may be repeated up to 5 times. The number of repetitions actually used will depend on the algorithm used. The order of the parameters is arbitrary but, in each case, the actual value is followed by a coded algorithm parameter Qualifier. Most algorithms in use today will not require parameter values to be the full allowable length.

Where the USA segment is used within a USH segment, the algorithm may be either symmetric, or a hash function.

Asymmetric algorithms shall not be referred to directly in USH segments but may appear only within Segment Group 2, triggered by a USC segment.

3.3.3 Segment Rules

3.3.3.1 SECURITY ALGORITHM (S502)

Use of algorithm, coded (0523)

Specifies the usage made of the algorithm identified by the algorithm data element (0527).

The use of algorithm is required to enable the different SECURITY ALGORITHM composite data elements, used in the different USA segments of Segment Group 2, to be distinguished from each other. In a USH segment the usage of the algorithm is determined by the Security Function qualifier of the USH segment. The use of algorithm **must** be used with one of the following codes :

Code	Mnemo	Meaning	Description
1	OHA	Owner hashing	Specifies that the algorithm is used by the message sender to compute the hash function on the message. (as in the case of Non-repudiation of Origin identified in the security function qualifier of USH).
2	OSY	Owner symmetric	Specifies that the algorithm is used by the message sender either for integrity or message origin authentication (specified by Security function qualifier in USH).

Cryptographic mode of operation, coded (0525)

Identifies the mode of operation used, for the algorithm specified by the algorithm (0527) data element.
In the USH segment one of the following codes **must** be used :

Code	Meaning	Description
0	NUL	Mode of operation meaningless for the current algorithm.
1	ECB	DES modes of operation, Electronic Code Book; FIPS Pub 81 (1981); ANSI X3.106; IS 8372 (64 bits); ISO 10116 (n-bits).
2	CBC	DES modes of operation, Cipher Block Chaining; FIPS Pub 81 (1981); ANSI X3.106; IS 8372 (64 bits); ISO 10116 (n-bits).
3	CFB1	DES modes of operation, Cipher feedback; FIPS Pub 81 (1981); ANSI X3.106; IS 8372 (64 bits); ISO 10116 (n-bits).
4	CFB8	DES modes of operation, Cipher feedback; FIPS Pub 81 (1981); ANSI X3.106; IS 8372 (64 bits); ISO 10116 (n-bits).
5	OFB	DES modes of operation. FIPS Pub 81 (1981); IS 8372 (64 bits); ISO 10116 (n-bits).
6	MAC	Message Authentication Code ISO 8731-1, using DES CBC mode.
7	DIM1	Data integrity mechanism using a cryptographic check function; ISO DIS 9797, first method
8	DIM2	Data integrity mechanism using a cryptographic check function; ISO DIS 9797, second method
9	MDC2	Modification Detection Code - IBM System Journal, vol 30, no 2, 1991.
10	HDS1	Hash functions - Part 2 : Hash functions using a n-bit block cipher algorithm providing a single length hash code. ISO CD 10118-2.
11	HDS2	Hash functions - Part 2 : Hash functions using a n-bit block cipher algorithm providing a double length hash code. ISO CD 10118-2.
12	SQM	Square-mod n hash function for RSA. Annex D, CCITT X 509. ISO 9594-8.
13	NVB7.1	Dutch Standard hash function for banking.
14	NVBAK	Dutch Banking Standard, NVB Authenticity Mark, published by the NVB, May 1992.

999	ZZZ	Mutually agreed.

Mode of operation code list identifier (0533)

Specification of the code lists used to identify the cryptographic mode of operation. When the codes defined above, by the UN/EDIFACT SJWG, as published in the present document, are used the value "1" **must** be used.

Algorithm, coded (0527)

Identifies the algorithm. In the USH segment one of the following codes **must** be used :

Code	Meaning	Description
1	DES	Data Encryption Standard. FIPS Pub 46 (January 1977).
2	MAA	Message Authentication Algorithm. Banking-Approved Algorithms for message Authentication. ISO 8731-2.
3	FEAL	FEAL Fast Data Encipherment Algorithm.
4	IDEA	International Data Encryption Algorithm : Lai X., Massey J. "A Proposal for a New Block Encryption Standard", Proceedings of Eurocrypt'90, LNCS vol 473, Springer-Verlag, Berlin 1991, and Lai X., Massey J. "Markov Ciphers and Differential Cryptanalysis", Proceedings of Eurocrypt'91, LNCS vol 547, Springer-Verlag, Berlin 1991.
5	MD4	The MD4 Message digest algorithm. Rivest R. RSA Data Security Inc. (1990).
6	MD5	The MD5 Message digest algorithm. Rivest R. Dusse S. RSA Data Security Inc. (1991).
7	RIPEMD	Extension of the MD4 - Ripe Report CS - R9324, April 93.
8	SHA	Secure Hashing Algorithm.
9	AR/DFP	Hash function of the German banking industry, submitted to ISO/IEC JTC 1/SC 27/WG 2, Doc N179.
999	ZZZ	Mutually agreed.

The presence of a particular algorithm in the list does not imply any endorsement of that algorithm.

The above algorithms are those currently in common use (November 1993). It is expected that new algorithms will be added as they become available, and are generally accepted.

Algorithm code list identifier (0529)

Specification of the code lists used to identify the algorithm. When the codes defined above by the UN/EDIFACT SJWG, as published in the present document, are used the value "1" **must** be used.

3.3.3.2 ALGORITHM PARAMETER (S503)

Algorithm parameter value (0532)

This component contains the value of a parameter required by the algorithm referenced in the algorithm data element. The precise type, usage and format of the value is specified in the immediately following algorithm parameter qualifier (0531). If necessary, this value is filtered by the filter function identified in the FILTER FUNCTION, CODED data element (0505) of the USH segment (key names do not need to be filtered).

Algorithm parameter qualifier (0531)

Identifies the type of the algorithm parameter value that immediately precedes it.

One of the following codes **must** be used :

Code	Mnemo	Meaning	Description
1	IVC	Initalisation Value, cleartext	Identifies the algorithm parameter value as an unencrypted initialisation value.
2	IVE	Initialisation Value, encrypted under a symmetric key	Identifies the algorithm parameter value as an initialisation value which is encrypted under the symmetric data key.
3	IVP	Initialisation Value, encrypted under a public key	Identifies the algorithm parameter value as an initialisation value encrypted under the public key of the receiving entity.
4	IVZ	Initialisation Value, format mutually agreed	Identifies the algorithm parameter value as an initialisation value in a format agreed between the two parties.
5	KYE	Symmetric key, encrypted under a symmetric key	Identifies the algorithm parameter value as a symmetric key which is encrypted with a previously agreed algorithm under a previously exchanged symmetric key.
6	KYP	Symmetric key, encrypted under a public key	Identifies the algorithm parameter value as a symmetric key encrypted under the public key of the receiving entity.
7	KYS	Symmetric key, signed and encrypted	Identifies the algorithm parameter value as a symmetric key signed under the sender's secret key, then encrypted under the receiver's public key.
8	KYD	Symmetric key encrypted under an asymmetric key common to the sender and the receiver	Identifies the algorithm parameter value as a symmetric key encrypted under an asymmetric key common to the sender and the receiver (use of Diffie and Hellmann scheme, for instance).
9	KYN	Symmetric key name	Identifies the algorithm parameter value as the name of a symmetric key. This may be used in the case where a key relationship has already been established between the sender and receiver.
10	KKN	Key encrypting key name	Identifies the parameter value as the name of a key encrypting key.
11	KYZ	Symmetric key, format mutually agreed	Identifies the algorithm parameter value as a symmetric key in a format agreed between the two parties.
999	ZZZ	Parameter value is mutually agreed	Identifies the algorithm parameter value as having a usage and format that is mutually agreed.

3.3.4 EXAMPLE

SEGMENT GROUP 2 (Conditional, 2)

USC	Certificate	M	1
USA	Security Algorithm	C	3
USR	Security Result	C	1

When asymmetric algorithms are used, this segment group contains the data necessary to validate the security methods applied to a message.

In its most common form, the certificate will include the public key and the credentials of the certificate owner signed by the Certificate Issuer.

A certificate may either be conveyed completely (the USC segment, 3 USA segments and the USR segment), or may be conveyed only as a Certificate Identifier (the USC segment identifying the Certificate), to refer to a public key that is already known by the entities involved or retrieved from a data base.

Two occurrences of this Segment group are allowed, one being the message sender certificate (that the message receiver will use to verify the message sender's signature), the other being the message receiver certificate (presumably only referred to by Certificate Reference) in the case where the receiver public key is used by the sender for confidentiality of symmetric keys.

If two certificates (sender's and receiver's) are simultaneously present within one security header, the Security Identification Details data element (S500) together with the Certificate Reference data element (0536) allow them to be differentiated.

3.4 USC - CERTIFICATE (Mandatory, 1)

3.4.1 Segment Format

Number	Description	M/C	Format	Special notes
0536	CERTIFICATE REFERENCE	C	an..35	
S500	SECURITY IDENTIFICATION DETAILS	C		
0577	Security party qualifier	M	an..3	
0538	Key name	C	an..35	
0511	Party Id identification	C	an..17	
0513	Code list qualifier	C	an..3	
0515	Code list responsible agency	C	an..3	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
S500	SECURITY IDENTIFICATION DETAILS	C		
0577	Security party qualifier	M	an..3	
0538	Key name	C	an..35	
0511	Party Id identification	C	an..17	
0513	Code list qualifier	C	an..3	
0515	Code list responsible agency	C	an..3	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
0586	Party name	C	an..35	
0544	FORMAT CERTIFICATE VERSION	C	an..3	
0505	FILTER FUNCTION, CODED	C	an..3	
0507	CHARACTER SET ENCODING, CODED	C	an..3	
0543	CHARACTER SET REPERTOIRE, CODED	C	an..3	
0546	USER AUTHORISATION LEVELS	C	an..35	
S505	SEPARATOR FOR SIGNATURE	C		
0548	Separator for signature	C	an..4	
0551	Separator for signature qualifier	C	an..3	
S505	SEPARATOR FOR SIGNATURE	C		
0548	Separator for signature	C	an..4	
0551	Separator for signature qualifier	C	an..3	
S505	SEPARATOR FOR SIGNATURE	C		
0548	Separator for signature	C	an..4	
0551	Separator for signature qualifier	C	an..3	
S505	SEPARATOR FOR SIGNATURE	C		
0548	Separator for signature	C	an..4	
0551	Separator for signature qualifier	C	an..3	
S501	SECURITY DATE AND TIME	C		
0515	Date and time qualifier, coded	M	an..3	
0502	Date	C	n8	
0504	Time	C	n6	
0506	UTC offset	C	an..5	
S501	SECURITY DATE AND TIME	C		
0515	Date and time qualifier, coded	M	an..3	
0502	Date	C	n8	
0504	Time	C	n6	
0506	UTC offset	C	an..5	
S501	SECURITY DATE AND TIME	C		
0515	Date and time qualifier, coded	M	an..3	
0502	Date	C	n8	
0504	Time	C	n6	
0506	UTC offset	C	an..5	

3.4.2 Segment Description

USC segments will be needed when public key cryptography is used, even if certificates are not used. Either the full certificate is present (including the USR segment), or the only data elements of the certificate are the Certificate reference (0536) and the SECURITY IDENTIFICATION DETAILS (S500) composite data element identifying the isuer Certification Authority or the SECURITY IDENTIFICATION DETAILS (S500) composite data element identifying the Certificate Owner, including its public key name. The presence of a full certificate may be avoided if the certificate has already been exchanged by the two parties.

3.4.3 Segment Rules

3.4.3.1 CERTIFICATE REFERENCE (0536)

Uniquely identifies one certificate for a Certification Authority. This field may be used to refer to a certificate when the whole certificate is not conveyed. The unique certificate reference may be obtained by the Certificate reference (0536) and the SECURITY IDENTIFICATION DETAILS (S500) composite data element identifying the Certification Authority.

3.4.3.2 SECURITY IDENTIFICATION DETAILS (S500)

Identification of parties involved in the certification process.

Two occurrences of this composite data element are possible : one for the Certificate Owner (identifying the entity which signs the message), one for the Certificate Issuer (Certification Authority or CA).

When this composite data element is used at least the Security Party Qualifier **must** be present.

The identification of Certificate Owner and Certification Authority is achieved by the data element Security Party Qualifier.

Security party qualifier (0577)

Specification of the function of the security party identified. One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
3	OW	Certificate Owner	Identifies the party which owns the Certificate.
4	AX	Authenticating party	Party which certifies that the document (i. e. the certificate) is authentic.

Key name (0538)

Identification of a public key : either the public key of the owner of this certificate, or the public key related to the secret key used by the Certificate Issuer (CA) to sign this certificate.
In the latter case, this field allows a Certification Authority to use several keys, either for separate applications, or consecutive generations of CA keys.

Party Id identification (0511)

Code identifying a party involved in the security process, according to a defined registry of security parties. In the USC segment this party may be either :

- the party which owns the Certificate (Certificate Owner), or
- the party which certifies that the document (i. e. the certificate) is authentic (Authenticating party : CA)

Code list qualifier (0513)

Code identifying the type of identification used to register the security parties.

Code list responsible agency (0515)

Code identifying the agency in charge of registration of the security parties.

Party name (0586)

Identification of the security party. Three occurences may be used to allow a complete identification.

3.4.3.3 FORMAT CERTIFICATE VERSION (0544)

Version number of the version of the certificate, identified by year and status of the UN/EDIFACT service segment directory.

3.4.3.4 FILTER FUNCTION, CODED (0505)

Identification of the filtering function used for validation results and keys, within the certificate.

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	NUL	No filter	Self explanatory
2	HEX	Hexadecimal filter	Hexadecimal filter
3	ASC	ISO 646 filter	ASCII filter as described in DIS 10126-1 (see note)
4	BAU	ISO 646 Baudot filter	Baudot filter as described in DIS 10126-1
5	EDA	UN/EDIFACT level A filter	UN/EDIFACT level A filter function as described in Annex A of the present document
999	ZZZ	Mutually agreed	Self explanatory

note :

This filtering function is mentioned here for completeness, but does not comply with the requirements of UN/EDIFACT level A or B syntax.

3.4.3.5 CHARACTER SET ENCODING, CODED (0507)

Identifies the character set in which the certificate was coded when security mechanisms were applied to it.

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description

1	AS7	ASCII 7 bit	Self explanatory
2	AS8	ASCII 8 bit	Self explanatory
3	EBC	EBCDIC IBM 360	IBM 360 machine EBCDIC code
4	IPC	ASCII IBM PC	IBM PC ASCII code
5	VAX	ASCII DEC VAX	DEC VAX ASCII 8 bit code
999	ZZZ	Mutually agreed	Self explanatory

3.4.3.6 CHARACTER SET REPERTOIRE, CODED (0543)

Identifies the syntax level used to create the certificate, when security mechanisms were applied to it.

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	UNOA	UN/EDIFACT syntax level A	Self explanatory
2	UNOB	UN/EDIFACT syntax level B	Self explanatory
3	UNOC	UN/EDIFACT syntax level C	Self explanatory
4	UNOD	UN/EDIFACT syntax level D	Self explanatory
5	UNOE	UN/EDIFACT syntax level E	Self explanatory
6	UNOF	UN/EDIFACT syntax level F	Self explanatory

3.4.3.7 USER AUTHORISATION LEVELS (0546)

Specification of the privileges, authorisation level, etc. associated with the owner of the certificate.

3.4.3.8 SEPARATOR FOR SIGNATURE (S505)

Identifies the characters used as syntactical separators between all components or within composite components, of the USC segment when the signature was computed. The syntactical separators used in the message may be different to these characters. The data element Separator for signature qualifier (0551) identifies each separator. If the composite data elements SEPARATOR FOR SIGNATURE (S505) are not present, the syntactical characters used are those currently used in the message.

Separator for signature (0548)

Separator used when the signature was computed. In order to avoid translator problems, this separator is represented by its value in the character set identified by the CHARACTER SET ENCODING data element (0507), hexa-filtered on, at least, two characters. For example the separator "'" is coded "27" (two characters), ":" is coded "3A" (two characters), the release character "?" is coded "3F" (two characters) and the separator "+" is coded "2B", if ASCII 8bit code page is used.

Separator for signature qualifier (0551)

Identifies each separator (either separator between data element or separator within a composite). One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
1	SEG	Segment separator	Specifies that this is the separator between segments.
2	DAT	Data separator	Specifies that this is the separator between data elements.
3	COM	Composite separator	Specifies that this is the separator within a composite data element.
4	REL	Release character	Specifies that this is the release character.

3.4.3.9 SECURITY DATE AND TIME (S501)

Identification of the dates and times involved in the certification process.

Three occurrences of this composite data element are possible : one for the certificate generation date and time, one for the certificate start of validity period , one for the certificate end of validity period.

The distinction between the certificate generation date and time, the certificate start of validity period and of the certificate end of validity period is achieved by the Date and time qualifier.

Date and time qualifier (0515)

One of the following codes **must** be used :

Code	Mnemo	Meaning	Description
2	CGT	Certificate generation time	Identifies the date and time of generation of the certificate by the Certification Authority.
3	CSV	Certificate start of validity period	Identifies the date and time from which the certificate must be considered valid.
4	CEV	Certificate end of validity period	Identifies the date and time until which the certificate must be considered valid.

Date (0502)

Specification of the date. Its format **must** be YYYYMMDD (century included).

Time (0504)

Specification of the time. Its format **must** be HHMMSS (HH being in 24 hour clock format).

UTC offset (0506)

Offset from UTC standard time. Its format may be either :

XHHMM (X being P for plus or M for minus, followed by the offset, in hours and minutes), or
XY (X being : C = Central, E = Eastern, M = Mountain, P = Pacific, and Y being : D = daylight time, S = Standard time, T = Time).

3.4.4 EXAMPLE

3.5 USA - SECURITY ALGORITHM (Mandatory, 3)

3.5.1 Segment Format

Number	Description	M/C	Format	Special notes
S502	SECURITY ALGORITHM	M		
0523	Use of algorithm, coded	M	an..3	
0525	Cryptographic mode of operation, coded	C	an..3	
0533	Mode of operation code list identifier	C	an..3	
0527	Algorithm, coded	C	an..3	
0531	Algorithm code list identifier	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier, coded	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier, coded	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier, coded	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier, coded	C	an..3	
S503	ALGORITHM PARAMETER	C		
0532	Algorithm parameter value	C	an..512	
0531	Algorithm parameter qualifier, coded	C	an..3	

3.5.2 Segment Description

This segment is used to identify an algorithm, the technical usage made of it, and to hold the technical parameters required.

The use of algorithm data element specifies the usage made of the algorithm, the algorithm data element identifies the algorithm itself and the cryptographic mode of operation data element specifies the mode of operation used.

The mode of operation code list identifier and the algorithm code list identifier identify the code list in which the codes of the preceding data elements (mode of operation or algorithm) are defined.

The algorithm parameters composite data element provides space for one parameter. It may be repeated up to 5 times. The number actually used will depend on the algorithm used. The order of the parameters is arbitrary but, in each case, the actual value is followed by a coded algorithm parameter qualifier. Most algorithms in use today will not require parameter values to be the full allowable length.

In Segment Group 2, triggered by the USC segment, three USA segments are present. These are :

1. the algorithm used by the Certificate Issuer to compute the hash value of the Certificate (hashing function)

2. the algorithm used by the Certificate Issuer to generate the Certificate (i. e. to sign the result of the hash function computed on the certificate content) (asymmetric algorithm)

3.a - either the algorithm used by the sender to sign the message (i. e. to sign the result of the hash function described in the USH segment, computed on the message content) (asymmetric algorithm),

3.b - or the receiver's asymmetric algorithm used by the sender to encrypt the key required by a symmetric algorithm applied to the message content and refered to by the Segment Group 1 triggered by the USH segment (asymmetric algorithm),

These three occurrences of the USA segment are distinguished by the Use of algorithm data element (0523).

3.5.3 Segment Rules

3.5.3.1 SECURITY ALGORITHM (S502)

Use of algorithm (0523)

Specifies the usage made of the algorithm identified by Algorithm data element (0527). In a USA segment within a USC one of the following codes **must** be used :

Code	Mnemo	Meaning	Description
3	ISG	Issuer signing	Specifies that the algorithm is used by the Certificate Issuer (CA) to sign the hash result computed on the certificate
4	IHA	Issuer hashing	Specifies that the algorithm is used by the Certificate Issuer (CA) to compute the hash result on the certificate
5	OCF	Owner enciphering	Specifies that the algorithm is used by the message sender to encrypt a symmetric key
6	OSG	Owner signing	Specifies that the algorithm is used by the message sender to sign either the hash result computed on the message or the symmetric keys
7	OCS	Owner enciphering or signing	Specifies that the algorithm is used by the message sender to encrypt a symmetric key or sign the hash result computed on the message

Cryptographic mode of operation, coded (0525)

Identifies the mode of operation used, for the algorithm specified by the algorithm (0527) data element.
In the USC segment one of the following codes **must** be used :

Code	Meaning	Description
0	NUL	Mode of operation meaningless for the current algorithm.
9	MDC2	Modification Detection Code - IBM System Journal, vol 30, no 2, 1991.
10	HDS1	Hash functions - Part 2 : Hash functions using a n-bit block cipher algorithm providing a single length hash code. ISO CD10118-2.
11	HDS2	Hash functions - Part 2 : Hash functions using a n-bit block cipher algorithm providing a double length hash code. ISO CD10118-2.
12	SQM	Square-mod n hash function for RSA. Annex D, CCITT X 509. ISO 9594-8.
15	MCCP	Banking key management by means of asymmetric algorithms, Algorithms using the RSA cryptosystem. Signature construction by means of a separate signature. ISO CD 11166-2.
16	DSMR	Digital Signature Scheme Giving message recovery. ISO 9796.
999	ZZZ	Mutually agreed.

Mode of operation code list identifier (0533)

Specification of the code lists used to identify the cryptographic mode of operation. When the codes defined above by the UN/EDIFACT SJWG, as published in the present document, are used the value "1" **must** be used.

Algorithm, coded (0527)

Specifies the algorithm. In the USC segment one of the following codes **must** be used :

Code	Meaning	Description
1	DES	Data Encryption Standard. FIPS Pub 46 (January 1977).
5	MD4	The MD4 Message digest algorithm. Rivest R. RSA Data Security Inc. (1990).
6	MD5	The MD5 Message digest algorithm. Rivest R. Dusse.S RSA Data Security Inc. (1991).
7	RIPEMD	Extension of the MD4 - Ripe Report CS - R9324, April 93.
8	SHA	Secure Hashing Algorithm.
9	AR/DFP	Hash function of the German banking industry, submitted to ISO/IEC JTC 1/SC 27/WG 2, Doc N179.
10	RSA	Rivest, Shamir, Adleman: A Method for obtaining Digital Signatures and Public Key Cryptosystems. Communications of the ACM, Vol.21(2), pp 120-126 (1978).
11	DSA	Digital Signature Algorithm/Digital Signature Standard NIST Pub 1993 Draft.
12	RAB	Rabin, "Digitalized signatures and public-key functions as intractable as factorization", MIT Laboratory for Computer Science Technical Report LCS/TR-212, Cambridge, Mass, 1979.
999	ZZZ	Mutually agreed.

Algorithm code list identifier (0529)

Specification of the code lists used to identify the algorithm. When the codes defined above by the UN/EDIFACT SJWG, as published in the present document, are used the value "1" **must** be used.

3.5.3.2 ALGORITHM PARAMETER (S503)

Algorithm parameter value (0532)

This component contains the value of a parameter required by the algorithm referenced in the algorithm data element. The precise type, usage and format of the value is specified in the immediately following algorithm parameter qualifier. If necessary, this value is filtered by the filter function identified in the FILTER FUNCTION data element (0505) of the USC segment (key names do not need to be filtered).

Algorithm parameter qualifier (0531)

Identifies the type of the algorithm parameter value that immediately precedes it.

One of the following codes **must** be used :

Code	Mnemo.	Meaning	Description
12	MOD	Modulus	Identifies the algorithm parameter value as the modulus of a public key which is to be used according to the function defined by the use of algorithm.
13	EXP	Exponent	Identifies the algorithm parameter value as the exponent of a public key which is to be used according to the function defined by the use of algorithm.
14	MLN	Modulus Length	Identifies the algorithm parameter value as the length of the modulus (in bits) of the public key used in the algorithm. The length is independent of whatever filtering function may be in use.
15	PR1	Generic parameter 1	Identifies the algorithm parameter value as the first generic parameter (see note)
16	PR2	Generic parameter 2	Identifies the algorithm parameter value as the second generic parameter (see note)
17	PR3	Generic parameter 3	Identifies the algorithm parameter value as the third generic parameter (see note)
18	PR4	Generic parameter 4	Identifies the algorithm parameter value as the fourth generic parameter (see note)
19	PR5	Generic parameter 5	Identifies the algorithm parameter value as the fifth generic parameter (see note)
20	PR6	Generic parameter 6	Identifies the algorithm parameter value as the sixth generic parameter (see note)
21	PR7	Generic parameter 7	Identifies the algorithm parameter value as the seventh generic parameter (see note)
22	PR8	Generic parameter 8	Identifies the algorithm parameter value as the eighth generic parameter (see note)
23	PR9	Generic parameter 9	Identifies the algorithm parameter value as the ninth generic parameter (see note)
24	PRA	Generic parameter 10	Identifies the algorithm parameter value as the tenth generic parameter (see note)
999	ZZZ	Parameter value is mutually agreed	Identifies the algorithm parameter value as having a usage and format that is mutually agreed.

note :
These generic parameters are provided to allow the use of any algorithm requiring identification of parameters different from the parameters defined above.

When the DSA algorithm (NIST, Pub 1993) is used, PR1 contains parameter "P", PR2 contains parameter "Q", PR3 contains parameter "G", PR4 contains parameter "Y".

3.5.4 EXAMPLE

3.6 USR - SECURITY RESULT (Mandatory, 1)

3.6.1 Segment Format

Number	Description	M/C	Format	Special notes
S508	VALIDATION RESULT	M		
0560	Validation value	M	an..256	
0560	Validation value	C	an..256	

3.6.2 Segment Description

The USR segment included in the USC segment contains the signature computed by the Certification Authority by signing the hash result computed on the data of the credentials.

In the case of signature algorithms requiring two parameters to express the result, the two data elements validation result are used in an order described in the documents about these algorithms.

3.6.3 Segment Rules

3.6.3.1 VALIDATION RESULT (S508)

Validation value (0560)

This component contains the digital signature computed by the Certification Authority on the data of the credentials. This signature is computed using first the hash function defined by the qualifier "issuer hashing" ("4") in the use of algorithm data element, then the asymmetric algorithm defined by the qualifier "issuer signing" ("3") in the use of algorithm data element.

The digital signature is computed according to the rules and with the parameters specified in the USC segment (CHARACTER SET ENCODING, SEPARATOR FOR SIGNATURE, CHARACTER SET REPERTOIRE). The signature computation starts with the first character of the USC segment (namely a "U") and ends with last character of the last USA segment (including the separator following this USA segment).

This data element is filtered, if necessary, by the filter function identified in the FILTER FUNCTION data element (0505) of the USC segment.

The length of this data element is determined by the length of the key (one of the Algorithm parameter data elements, qualified by the Algorithm parameter qualifier "modulus length" ("14"), of the Issuer signature algorithm) and the filter function applied to the result of the signature process.

In the case of RSA signature, only one validation value data element is used.

In the case of DSA signature two validation value data elements are required. The first one corresponds to the parameter known as "r", the second one to the parameter known as "s".

3.6.4 EXAMPLE

3.7 USB, BEGINNING OF A SECURITY MESSAGE (Mandatory, 1)

3.7.1 SEGMENT FORMAT

Number	Description	M/C	Format	Special notes
0563	MESSAGE FUNCTION, CODED	C	an..3	
0503	RESPONSE TYPE, CODED	C	an..3	
S501	SECURITY DATE AND TIME	C		
0517	Date and time qualifier, coded	M	an..3	
0502	Date	C	n8	
0504	Time	C	n6	
0506	UTC offset	C	an..5	
0590	RELATED FILENAME	C	an..256	
S002	INTERCHANGE SENDER	C		
0004	Sender identification	C	an..35	
0007	Partner identification code qualifier	M	an..35	
0008	Address for reverse routing	C	an..14	
S003	INTERCHANGE RECIPIENT	C		
0010	Recipient identification	C	an..35	
0007	Partner identification code qualifier	C	an..14	
0014	Routing address	C	an..14	

3.7.2 Segment Description

A segment for identification of the AUTACK message, its type, function: authentication or acknowledgement, and related details.

3.7.3 Segment rules

3.7.3.1 MESSAGE FUNCTION, CODED (0563)

One of the following codes must be used:

Code	Mnemo.	Meaning	Description
1	AUT	Authentication	The message is used for authentication and/or non-repudiation of origin
2	ACK	Acknowledgement	The message is used to acknowledge receipt of messages or interchanges
3	NA	Non-Acknowledgement	The message is used to refuse acknowledgement, and mention security errors.

3.7.3.2 RESPONSE TYPE, CODED (0503)

It specifies whether a secure acknowledgment from the message recipient is required or not. If it is required, the message sender will expect an AUTACK message to be sent back by the current message recipient to the current message sender, containing this acknowledgement.

One of the following codes must be used:

Code	Mnemo.	Meaning	Description
1	NA	No acknowledgement required	No AUTACK acknowledgment message expected
2	AC	Acknowledgement	AUTACK acknowledgment message expected

3.7.3.3 SECURITY DATE AND TIME (S501)

Security timestamp of the message to which security is applied. This timestamp is security related and may differ from any dates and times that may appear somewhere else in the message. It may be used to provide message sequence integrity.

Date and time qualifier (0517)

Specification of the type of date and time the folowing code must be used.

Code	Mnemo.	Meaning	Description
5	MGT	Message generation d/t	Date and time at which the secured entity was generated

Date (0502)

Specification of the date. Its format must be YYYYMMDD (century included).

Time (0504)

Specification of the time. Its format must be HHMMSS (HH being in 24 clock).

UTC offset (0506)

Offset from UTC standard time. Its format may be cether:

XHHMM (X being P for plus or M for minus, followed by the offset, in hours and minutes), or

XY (X being: C = central, E = Eastern, M = Mountain, P = Pacific, and Y being: D = Daylight time, S= Standard time, T = Time.

3.7.3.4 RELATED FILENAME (0590)

This segment contains a filename for which the digital signature is placed in group 3.

3.7.3.5 INTERCHANGE SENDER (S002)

Sender identification (0004)

Name or coded representation of the sender of a data interchange.

Partner identification code qualifier (0007).
Qualifier referring to the source of codes for the identifiers of interchanging partners.

Address for reverse routing (0008)

Address specified by the sender of an interchange to be included by the recipient in the response interchanges to facilitate internal routing.

3.7.3.6 INTERCHANGE RECIPIENT (S003)

Recipient identification (0010)
Name or coded representation of the recipient of a data interchange.

Partner identification code qualifier (0007)
Qualifier referring to the source of codes for the identifiers of interchanging partners.

Routing address (0014)
Address specified by the recipient of an interchange to be included by the sender and used by the recipient for routing of received interchanges inside his organization.

3.7.4 EXAMPLE

Segment group 3 (Mandatory, 9999)

USX	Security references	M	1
USY	Security on references	C	9

This segment group identifies the messages or interchanges being secured by the present AUTACK.

3.8 USX, SECURITY REFERENCES (Mandatory, 1)

3.8.1 Segment Format

Number	Description	M/C	Format	Special notes
0020	INTERCHANGE CONTROL REFERENCE	M	an..14	
0062	MESSAGE REFERENCE NUMBER	C	an..14	
S501	SECURITY DATE AND TIME	C		
0157	Date and time qualifier, coded	M	an..8	
0502	Date	C	n8	
0504	Time	C	n6	
0506	UTC offset	C	an..5	

3.8.2 Segment Description

This segment is used to identify a message or an interchange in the security process.

3.8.3 Segment Rules

3.8.3.1 INTERCHANGE CONTROL REFERENCE (0020)

Contains the unique reference assigned by the sender to an interchange.

3.8.3.2 MESSAGE REFERENCE NUMBER (0062)

Contains the unique message reference number assigned by the sender.

3.8.3.3 SECURITY DATE AND TIME (S501)

Original generation date and time of the referenced message or interchange.

Date and time qualifier (0517)

Specification of the type of date and time. The following code must be used:

Code	Mnemo.	Meaning	Description
5	MGT	Message generation d/t	Date and time at which the secured entity was generated.

Date (0502)

Specification of the date. Its format must be YYYYMMDD (century included).

Time (0504)

Specification of the time. Its format must be HHMMSS (HH being in 24 clock).

UTC offset (0506)

Offset from UTC standard time. Its format may be cether:

XHHMM (X being P for plus or M for minus, followed by the offset, in hours and minutes), or

XY (X being: C = Central, E = Eastern, M = Mountain, P = Pacific, and Y being: D = Daylight time, S = Standard time, T = Time).

3.9 USY, SECURITY ON REFERENCES (Conditional, 9)

3.9.1 Segment Format

Number	Description	M/C	Format	Special notes
0534	SECURITY RESULT LINK	M	n2	
S508	VALIDATION RESULT	C		
0560	Validation value	M	an..256	
0560	Validation value	C	an..256	
0571	SECURITY ERROR, CODED	C	an..3	

3.9.2 Segment Description

This segment gives security information on the refered message or interchange.

3.9.3 Segment Rules

3.9.3.1 SECURITY RESULT LINK (0534)

Contains a number which links the validation result to the corresponding USH segment using the security functions, INT, RNO, RAU or RINT.
With the security function NRR or AUR the link is to the number of the original signature of the referenced message.

3.9.3.2 VALIDATION RESULT (S508)

Number	Description	M/C	Format	Special notes
S508	VALIDATION RESULT	C		
0560	Validation value	M	an..256	
0560	Validation value	C	an..256	

Contains the security result corresponding to the security functions specified in the linked USH segment, depending on the security function.

Validation value (0560)

This data element is filtered, if necessary, by the filter function identified in the FILTER FUNCTION data element (0505) of the USH segment.

The length of this data element is determined by the length of the key (one of the Algorithm parameter data elements, qualified by the Algorithm parameter qualifier "modulus length" ("14"), of the Owner signature algorithm) and the filter function applied to the result of the signature process.

In the case of RSA signature, only one validation value data element is used.

In the case of DSA signature two validation value data elements are required. The first one corresponds to the parameter known as "r", the second one to the parameter known as "s".

3.9.3.3 SECURITY ERROR, CODED (0571)

One of the following codes must be used.

Code	Mnemo.	Meaning	Description
0	NULL	No error	
1	NAUT	Wrong authenticator	The validation value is wrong
2	NCER	Wrong certificate	The certificate is wrong
3	NPATH	Certification path	The certifcation path is incomplete. Cannot verify.

4	NALG	Not supported	The algorithm is not supported
5	NHASH	Not supported	Hashing method is not supported
999	NDOC	Not documented	

3.9.4 EXAMPLE

SEGMENT GROUP 4 (Conditional, 9)

	Segment Group 3	C	9
UST	Security Trailer	M	1
USR	Security Result	C	1

This segment group contains the link with the related USH segment and the security result corresponding to the security functions specified in this USH segment. For every security trailer (Segment Group n, triggered by UST) there is one corresponding security header (Segment Group 1, triggered by USH).

3.10 UST - SECURITY TRAILER (Mandatory, 1)

3.10.1 Segment Format

Number	Description	M/C	Format	Special notes
0534	SECURITY RESULT LINK	M	n2	

3.10.2 Segment Description

This segment is used to separate the UNSM body from the security trailer.

The UST segment contains a number which links the UST segment with its corresponding USH segment.

3.10.3 Segment Rules

3.10.3.1 SECURITY RESULT LINK (0534)

Contains a number which links a particular USH segment with its corresponding UST segment. The value used is arbitrarily assigned but, within one message, the same value **must** not be used more than once.

3.10.4 EXAMPLE

3.11 USR - SECURITY RESULT (Conditional, 1)

3.11.1 Segment Format

Number	Description	M/C	Format	Special notes
S508	VALIDATION RESULT	M		
0560	Validation value	M	an..256	
0560	Validation value	C	an..256	

3.11.2 Segment Description

Contains the security result corresponding to the security functions specified in the linked USH segment

In the case of signature algorithms requiring two parameters to express the result, the two data elements validation result are used in an order described in the documents about these algorithms.

3.11.3 Segment Rules

3.11.3.1 VALIDATION RESULT (S508)

Validation value (0560)

Contains the security result corresponding to the security functions specified in the linked USH segment.

This data element is filtered, if necessary, by the filter function identified in the FILTER FUNCTION data element (0505) of the USH segment.

The length of this data element is determined by the length of the key (one of the Algorithm parameter data elements, qualified by the Algorithm parameter qualifier "modulus length" ("14"), of the Owner signature algorithm) and the filter function applied to the result of the signature process.

In the case of RSA signature, only one validation value data element is used.

In the case of DSA signature two validation value data elements are required. The first one corresponds to the parameter known as "r", the second one to the parameter known as "s".

3.11.4 EXAMPLE

4. HOW TO PROTECT AN EDIFACT MESSAGE

4.1 AUTACK as authenticity message

4.2 AUTACK as non-repudiation of receipt

5. MESSAGE PROTECTION EXAMPLES

5.1 AUTACK as authenticity message

5.2 AUTACK as non-repudiation of receipt

EXHIBIT B-5

RFC 1505

Network Working Group
Request for Comments: 1505
Obsoletes: 1154

A. Costanzo
AKC Consulting
D. Robinson
Computervision Corporation
R. Ullmann
August 1993

Encoding Header Field for Internet Messages

Status of this Memo

This memo defines an Experimental Protocol for the Internet community. It does not specify an Internet standard. Discussion and suggestions for improvement are requested. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

IESG Note

Note that a standards-track technology already exists in this area [11].

Abstract

This document expands upon the elective experimental Encoding header field which permits the mailing of multi-part, multi-structured messages. It replaces RFC 1154 [1].

Table of Contents

1. Introduction

STD 11, RFC 822 [2] defines an electronic mail message to consist of two parts, the message header and the message body, separated by a blank line.

The Encoding header field permits the message body itself to be further broken up into parts, each part also separated from the next by a blank line. Thus, conceptually, a message has a header part, followed by one or more body parts, all separated by apparently blank lines. Each body part has an encoding type. The default (no Encoding field in the header) is a one part message body of type "Text".

The purpose of Encoding is to be descriptive of the content of a mail message without placing constraints on the content or requiring additional structure to appear in the body of the message that will interfere with other processing.

A similar message format is used in the network news facility, and posted articles are often transferred by gateways between news and mail. The Encoding field is perhaps even more useful in news, where articles often are uuencoded or shar'd, and have a number of different nested encodings of graphics images and so forth. In news in particular, the Encoding header keeps the structural information within the (usually concealed) article header, without affecting the visual presentation by simple news-reading software.

2. The Encoding Field

The Encoding field consists of one or more subfields, separated by commas. Each subfield corresponds to a part of the message, in the order of that part's appearance. A subfield consists of a line count and a keyword or a series of nested keywords defining the encoding. The line count is optional in the last subfield.

2.1 Format of the Encoding Field

The format of the Encoding field is:

```
[  <count> <keyword> [ <keyword> ]* ,  ]*
        [ <count> ] <keyword> [ <keyword> ]*
```

where:

```
<count>    := a decimal integer
<keyword>  := a single alphanumeric token starting with an alpha
```

2.2 <count>

The line count is a decimal number specifying the number of text lines in the part. Parts are separated by a blank line, which is not included in the count of either the preceding or following part. Blank lines consist only of CR/LF. Count may be zero, it must be non-negative.

It is always possible to determine if the count is present because a count always begins with a digit and a keyword always begins with a letter.

The count is not required on the last or only part. A multi-part message that consists of only one part is thus identical to a single-part message.

2.3 <keyword>

Keyword defines the encoding type. The keyword is a common single-word name for the encoding type and is not case-sensitive.

```
Encoding: 107 Text
```

2.3.1 Nested Keywords

Nested keywords are a series of keywords defining a multi-encoded message part. The encoding keywords may either be an actual series of encoding steps the encoder used to generate the message part or may merely be used to more precisely identify the type of encoding (as in the use of the keyword "Signature").

Nested keywords are parsed and generated from left to right. The order is significant. A decoding application would process the list from left to right, whereas, an encoder would process the Internet message and generate the nested keywords in the reverse order of the actual encoding process.

```
Encoding: 458 uuencode LZW tar (Unix binary object)
```

2.4 Comments

Comments enclosed in parentheses may be inserted anywhere in the encoding field. Mail reading systems may pass the comments to their clients. Comments must not be used by mail reading systems for content interpretation. Other parameters defining the type of encoding must be contained within the body portion of the Internet message or be implied by a keyword in the encoding field.

3. Encodings

This section describes some of the defined encodings used. An alphabetical listing is provided in Section 6.

As with the other keyword-defined parts of the header format standard, new keywords are expected and welcomed. Several basic principles should be followed in adding encodings. The keyword should be the most common single word name for the encoding, including acronyms if appropriate. The intent is that different implementors will be likely to choose the same name for the same encoding. Keywords should not be too general: "binary" would have been a bad choice for the "hex" encoding.

The encoding should be as free from unnecessary idiosyncracies as possible, except when conforming to an existing standard, in which case there is nothing that can be done.

The encoding should, if possible, use only the 7 bit ASCII printing characters if it is a complete transformation of a source document (e.g., "hex" or "uuencode"). If it is essentially a text format, the full range may be used. If there is an external standard, the character set may already be defined. Keywords beginning with "X-" are permanently reserved to implementation-specific use. No standard registered encoding keyword will ever begin with "X-".

New encoding keywords which are not reserved for implementation-specific use must be registered with the Internet Assigned Numbers Authority (IANA). Refer to section 3.17 for additional information.

3.1 Text

This indicates that the message is in no particular encoded format, but is to be presented to the user as-is.

The text is ISO-10646-UTF-1 [3]. As specified in STD 10, RFC 821 [10], the message is expected to consist of lines of reasonable length (less than or equal to 1000 characters).

On some older implementations of mail and news, only the 7 bit subset of ISO-10646-UTF-1 can be used. This is identical to the ASCII 7 bit code. On some mail transports that are not compliant with STD 10, RFC 821 [10], line length may be restricted by the service.

Text may be followed by a nested keyword to define the encoded part further, e.g., "signature":

```
Encoding: 496 Text, 8 Text Signature
```

An automated file sending service may find this useful, for example, to differentiate between and ignore the signature area when parsing the body of a message for file requests.

3.2 Message

This encoding indicates that the body part is itself in the format of an Internet message, with its own header part and body part(s). A "message" body part's message header may be a full Internet message header or it may consist only of an Encoding field.

Using the message encoding on returned mail makes it practical for a mail reading system to implement a reliable automatic resending function, if the mailer generates it when returning contents. It is also useful in a "copy append" MUA (mail user agent) operation.

MTAs (mail transfer agents) returning mail should generate an Encoding header. Note that this does not require any parsing or transformation of the returned message; the message is simply appended un-modified; MTAs are prohibited from modifying the content of messages.

Encoding: 7 Text (Return Reason), Message (Returned Mail)

3.3 Hex

The encoding indicates that the body part contains binary data, encoded as 2 hexadecimal digits per byte, highest significant nibble first.

Lines consist of an even number of hexadecimal digits. Blank lines are not permitted. The decode process must accept lines with between 2 and 1000 characters, inclusive.

The Hex encoding is provided as a simple way of providing a method of encoding small binary objects.

3.4 EVFU

EVFU (electronic vertical format unit) specifies that each line begins with a one-character "channel selector". The original purpose was to select a channel on a paper tape loop controlling the printer.

This encoding is sometimes called "FORTRAN" format. It is the default output format of FORTRAN programs on a number of computer systems.

The legal characters are '0' to '9', '+', '-', and space. These correspond to the 12 rows (and absence of a punch) on a printer control tape (used when the control unit was electromechanical).

The channels that have generally agreed definitions are:

1	advances to the first print line on the next page
0	skip a line, i.e., double-space
+	over-print the preceeding line
-	skip 2 lines, i.e., triple-space
(space)	print on the next line, single-space

3.5 EDI-X12 and EDIFACT

The EDI-X12 and EDIFACT keywords indicate that the message or part is a EDI (Electronic Document Interchange) business document, formatted according to ANSI X12 or the EDIFACT standard.

A message containing a note and 2 X12 purchase orders might have an encoding of:

```
Encoding: 17 TEXT, 146 EDI-X12, 69 EDI-X12
```

3.6 FS

The FS (File System) keyword specifies a section consisting of encoded file system objects. This encoding method (defined in section 4) allows the moving of a structured set of files from one environment to another while preserving all common elements.

3.7 LZJU90

The LZJU90 keyword specifies a section consisting of an encoded binary or text object. The encoding (defined in section 5) provides both compression and representation in a text format.

3.8 LZW

The LZW keyword specifies a section consisting of the data produced by the Unix compress program.

3.9 UUENCODE

The uuencode keyword specifies a section consisting of the output of the uuencode program supplied as part of uucp.

3.10 PEM and PEM-Clear

The PEM and PEM-Clear keywords indicate that the section is encrypted with the methods specified in RFCs 1421-1424 [4,5,6,7] or uses the MIC-Clear encapsulation specified therein.

A simple text object encrypted with PEM has the header:

```
Encoding: PEM Text
```

Note that while this indicates that the text resulting from the PEM decryption is ISO-10646-UTF-1 text, the present version of PEM further restricts this to only the 7 bit subset. A future version of PEM may lift this restriction.

If the object resulting from the decryption starts with Internet message header(s), the encoding is:

```
Encoding: PEM Message
```

This is useful to conceal both the encoding within and the headers not needed to deliver the message (such as Subject:).

PEM does not provide detached signatures, but rather provides the MIC-Clear mode to send messages with integrity checks that are not encrypted. In this mode, the keyword PEM-Clear is used:

```
Encoding: PEM-Clear EDIFACT
```

The example being a non-encrypted EDIFACT transaction with a digital signature. With the proper selection of PEM parameters and environment, this can also provide non-repudiation, but it does not provide confidentiality.

Decoders that are capable of decrypting PEM treat the two keywords in the same way, using the contained PEM headers to distinguish the mode. Decoders that do not understand PEM can use the PEM-Clear keyword as a hint that it may be useful to treat the section as text, or even continue the decode sequence after removing the PEM headers.

When Encoding is used for PEM, the RFC934 [9] encapsulation specified in RFC1421 is not used.

3.11 PGP

The PGP keyword indicates that the section is encrypted using the Pretty Good Privacy specification, or is a public key block, keyring, or detached signature meaningful to the PGP program. (These objects

are distinguished by internal information.)

The keyword actually implies 3 different transforms: a compression step, the encryption, and an ASCII encoding. These transforms are internal to the PGP encoder/decoder. A simple text message encrypted with PGP is specified by:

Encoding: PGP Text

An EDI transaction using ANSI X12 might be:

Encoding: 176 PGP EDI-X12

Since an evesdropper can still "see" the nested type (Text or EDI in these examples), thus making information available to traffic analysis which is undesirable in some applications, the sender may prefer to use:

Encoding: PGP Message

As discussed in the description of the Message keyword, the enclosed object may have a complete header or consist only of an Encoding: header describing its content.

When PGP is used to transmit an encoded key or keyring, with no object significant to the mail user agent as a result of the decoding (e.g., text to display), the keyword is used by itself.

Another case of the PGP keyword occurs in "clear-signing" a message. That is, sending an un-encrypted message with a digital signature providing authentication and (in some environments) non-deniability.

Encoding: 201 Text, 8 PGP Signature, 4 Text Signature

This example indicates a 201 line message, followed by an 8 line (in its encoded form) PGP detached signature. The processing of the PGP section is expected (in this example) to result in a text object that is to be treated by the receiver as a signature, possibly something like:

[PGP signed Ariel@Process.COM Robert L Ullmann VALID/TRUSTED]

Note that the PGP signature algorithm is applied to the encoded form of the clear-text section, not the object(s) before encoding. (Which would be quite difficult for encodings like tar or FS). Continuing the example, the PGP signature is then followed by a 4 line "ordinary" signature section.

3.12 Signature

The signature keyword indicates that the section contains an Internet message signature. An Internet message signature is an area of an Internet message (usually located at the end) which contains a single line or multiple lines of characters. The signature may comprise the sender's name or a saying the sender is fond of. It is normally inserted automatically in all outgoing message bodies. The encoding keyword "Signature" must always be nested and follow another keyword.

```
Encoding: 14 Text, 3 Text Signature
```

A usenet news posting program should generate an encoding showing which is the text and which is the signature area of the posted message.

3.13 TAR

The tar keyword specifies a section consisting of the output of the tar program supplied as part of Unix.

3.14 PostScript

The PostScript keyword specifies a section formatted according to the PostScript [8] computer program language definition. PostScript is a registered trademark of Adobe Systems Inc.

3.15 SHAR

The SHAR keyword specifies a section encoded in shell archive format. Use of shar, although supported, is not recommended.

WARNING: Because the shell archive may contain commands you may not want executed, the decoder should not automatically execute decoded shell archived statements. This warning also applies to any future types that include commands to be executed by the receiver.

3.16 Uniform Resource Locator

The URL keyword indicates that the section consists of zero or more references to resources of some type. URL provides a facility to include by reference arbitrary external resources from various sources in the Internet. The specification of URL is a work in progress in the URI working group of the IETF.

3.17 Registering New Keywords

New encoding keywords which are not reserved for implementation-specific use must be registered with the Internet Assigned Numbers Authority (IANA). IANA acts as a central registry for these values. IANA may reject or modify the keyword registration request if it does not meet the criteria as specified in section 3. Keywords beginning with "X-" are permanently reserved to implementation-specific use. IANA will not register an encoding keyword that begins with "X-". Registration requests should be sent via electronic mail to IANA as follows:

```
To: IANA@isi.edu
Subject: Registration of a new EHF-MAIL Keyword
```

The mail message must specify the keyword for the encoding and acronyms if appropriate. Documentation defining the keyword and its proposed purpose must be included. The documentation must either reference an external non-Internet standards document or an existing or soon to be RFC. If applicable, the documentation should contain a draft version of the future RFC. The draft must be submitted as a RFC according to the normal procedure within a reasonable amount of time after the keyword's registration has been approved.

4. FS (File System) Object Encoding

The file system encoding provides a standard, transportable encoding of file system objects from many different operating systems. The intent is to allow the moving of a structured set of files from one environment to another while preserving common elements. At the same time, files can be moved within a single environment while preserving all attributes.

The representations consist of a series of nested sections, with attributes defined at the appropriate levels. Each section begins with an open bracket "[" followed by a directive keyword and ends with a close bracket "]". Attributes are lines, beginning with a keyword. Lines which begin with a LWSP (linear white space) character are continuation lines.

Any string-type directive or attribute may be a simple string not starting with a quotation mark (") and not containing special characters (e.g. newline) or LWSP (space and tab). The string name begins with the first non-LWSP character on the line following the attribute or directive keyword and ends with the last non-LWSP character.

Otherwise, the character string name is enclosed in quotes. The string itself contains characters in ISO-10646-UTF-1 but is quoted and escaped at octet level (as elsewhere in RFC822 [2]). The strings begin and end with a quotation mark ("). Octets equal to quote in the string are escaped, as are octets equal to the escape characters (\" and \\). The escaped octets may be part of a UTF multi-octet character. Octets that are not printable are escaped with \nnn octal representation. When an escape (\) occurs at the end of a line, the escape, the end of the line, and the first character of the next line, which must be one of the LWSP characters, are removed (ignored).

```
 [ file Simple-File.Name

 [ file "   Long file name starting with spaces and having a couple\
   [sic] of nasties in it like this newline\012near the end."
```

Note that in the above example, there is one space (not two) between "couple" and "[sic]". The encoder may choose to use the nnn sequence for any character that might cause trouble. Refer to section 5.1 for line length recommendations.

4.1 Sections

A section starts with an open bracket, followed by a keyword that defines the type of section.

The section keywords are:

```
directory
entry
file
segment
data
```

The encoding may start with either a file, directory or entry. A directory section may contain zero or more file, entry, and directory sections. A file section contains a data section or zero or more segment sections. A segment section contains a data section or zero or more segment sections.

4.1.1 Directory

This indicates the start of a directory. There is one parameter, the entry name of the directory:

```
[ directory foo
...
]
```

4.1.2 Entry

The entry keyword represents an entry in a directory that is not a file or a sub-directory. Examples of entries are soft links in Unix, or access categories in Primos. A Primos access category might look like this:

```
[ entry SYS.ACAT
type ACAT
created 27 Jan 1987 15:31:04.00
acl SYADMIN:* ARIEL:DALURWX $REST:
]
```

4.1.3 File

The file keyword is followed by the entry name of the file. The section then continues with attributes, possibly segments, and then data.

```
[ file MY.FILE
created 27 Feb 1987 12:10:20.07
modified 27 Mar 1987 16:17:03.02
type DAM
[ data LZJU90
* LZJU90
...
]]
```

4.1.4 Segment

This is used to define segments of a file. It should only be used when encoding files that are actually segmented. The optional parameter is the number or name of the segment.

When encoding Macintosh files, the two forks of the file are treated as segments:

```
[ file A.MAC.FILE
display "A Mac File"
type MAC
comment "I created this myself"
...
[ segment resource
[ data ...
...
]]
[ segment data
[ data ...
...
]]]
```

4.1.5 Data

The data section contains the encoded data of the file. The encoding method is defined in section 5. The data section must be last within the containing section.

4.2 Attributes

Attributes may occur within file, entry, directory, and segment sections. Attributes must occur before sub-sections.

The attribute directives are:

```
display
type
created
modified
accessed
owner
group
acl
password
block
record
application
```

4.2.1 Display

This indicates the display name of the object. Some systems, such as the Macintosh, use a different form of the name for matching or uniqueness.

4.2.2 Comment

This contains an arbitrary comment on the object. The Macintosh stores this attribute with the file.

4.2.3 Type

The type of an object is usually of interest only to the operating system that the object was created on.

Types are:

ACAT	access category (Primos)
CAM	contiguous access method (Primos)
DAM	direct access method (Primos)
FIXED	fixed length records (VMS)
FLAT	`flat file', sequence of bytes (Unix, DOS, default)
ISAM	indexed-sequential access method (VMS)
LINK	soft link (Unix)
MAC	Macintosh file
SAM	sequential access method (Primos)
SEGSAM	segmented direct access method (Primos)
SEGDAM	segmented sequential access method (Primos)
TEXT	lines of ISO-10646-UTF-1 text ending with CR/LF
VAR	variable length records (VMS)

4.2.4 Created

Indicates the creation date of the file. Dates are in the format defined in section 4.3.

4.2.5 Modified

Indicates the date and time the file was last modified or closed after being open for write.

4.2.6 Accessed

Indicates the date and time the file was last accessed on the original file system.

4.2.7 Owner

The owner directive gives the name or numerical ID of the owner or creator of the file.

4.2.8 Group

The group directive gives the name(s) or numerical IDs of the group or groups to which the file belongs.

4.2.9 ACL

This directive specifies the access control list attribute of an object (the ACL attribute may occur more than once within an object). The list consist of a series of pairs of IDs and access codes in the format:

```
user-ID:access-list
```

There are four reserved IDs:

```
$OWNER  the owner or creator
$GROUP  a member of the group or groups
$SYSTEM a system administrator
$REST   everyone else
```

The access list is zero or more single letters:

```
A    add (create file)
D    delete
L    list (read directory)
P    change protection
R    read
U    use
W    write
X    execute
*    all possible access
```

4.2.10 Password

The password attribute gives the access password for this object. Since the content of the object follows (being the raison d'etre of the encoding), the appearance of the password in plain text is not considered a security problem. If the password is actually set by the decoder on a created object, the security (or lack) is the responsibility of the application domain controlling the decoder as is true of ACL and other protections.

4.2.11 Block

The block attribute gives the block size of the file as a decimal number of bytes.

4.2.12 Record

The record attribute gives the record size of the file as a decimal number of bytes.

4.2.13 Application

This specifies the application that the file was created with or belongs to. This is of particular interest for Macintosh files.

4.3 Date Field

Various attributes have a date and time subsequent to and associated with them.

4.3.1 Syntax

The syntax of the date field is a combination of date, time, and timezone:

```
DD Mon YYYY HH:MM:SS.FFFFFF [+-]HHMMSS

Date :=  DD Mon YYYY      1 or 2 Digits " " 3 Alpha " " 4 Digits
DD   :=  Day              e.g. "08", " 8", "8"
Mon  :=  Month            "Jan" | "Feb" | "Mar" | "Apr" |
                          "May" | "Jun" | "Jul" | "Aug" |
                          "Sep" | "Oct" | "Nov" | "Dec"
YYYY :=  Year
Time :=  HH:MM:SS.FFFFFF  2 Digits ":" 2 Digits [ ":" 2 Digits
                          ["." 1 to 6 Digits ] ]
                          e.g. 00:00:00, 23:59:59.999999
HH   :=  Hours            00 to 23
MM   :=  Minutes          00 to 59
SS   :=  Seconds          00 to 60 (60 only during a leap second)
FFFFF:=  Fraction
Zone :=  [+-]HHMMSS       "+" | "-" 2 Digits [ 2 Digits
                          [ 2 Digits ] ]
HH   :=  Local Hour Offset
MM   :=  Local Minutes Offset
SS   :=  Local Seconds Offset
```

4.3.2 Semantics

The date information is that which the file system has stored in regard to the file system object. Date information is stored differently and with varying degrees of precision by different computer file systems. An encoder must include as much date information as it has available concerning the file system object. A

decoder which receives an object encoded with a date field containing greater precision than its own must disregard the excessive information. Zone is Co-ordinated Universal Time "UTC" (formerly called "Greenwich Mean Time"). The field specifies the time zone of the file system object as an offset from Universal Time. It is expressed as a signed [+-] two, four or six digit number.

A file that was created April 15, 1993 at 8:05 p.m. in Roselle Park, New Jersey, U.S.A. might have a date field which looks like:

15 Apr 1993 20:05:22.12 -0500

5. LZJU90: Compressed Encoding

LZJU90 is an encoding for a binary or text object to be sent in an Internet mail message. The encoding provides both compression and representation in a text format that will successfully survive transmission through the many different mailers and gateways that comprise the Internet and connected mail networks.

5.1 Overview

The encoding first compresses the binary object, using a modified LZ77 algorithm, called LZJU90. It then encodes each 6 bits of the output of the compression as a text character, using a character set chosen to survive any translations between codes, such as ASCII to EBCDIC. The 64 six-bit strings 000000 through 111111 are represented by the characters "+", "-", "0" to "9", "A" to "Z", and "a" to "z". The output text begins with a line identifying the encoding. This is for visual reference only, the "Encoding:" field in the header identifies the section to the user program. It also names the object that was encoded, usually by a file name.

The format of this line is:

* LZJU90 <name>

where <name> is optional. For example:

* LZJU90 vmunix

This is followed by the compressed and encoded data, broken into lines where convenient. It is recommended that lines be broken every 78 characters to survive mailers than incorrectly restrict line length. The decoder must accept lines with 1 to 1000 characters on each line. After this, there is one final line that gives the number of bytes in the original data and a CRC of the original data. This

should match the byte count and CRC found during decompression.

This line has the format:

* <count> <CRC>

where <count> is a decimal number, and CRC is 8 hexadecimal digits. For example:

* 4128076 5AC2D50E

The count used in the Encoding: field in the message header is the total number of lines, including the start and end lines that begin with *. A complete example is given in section 5.3.2.

5.2 Specification of the LZJU90 compression

The Lempel-Ziv-Storer-Szymanski model of mixing pointers and literal characters is used in the compression algorithm. Repeat occurrences of strings of octets are replaced by pointers to the earlier occurrence.

The data compression is defined by the decoding algorithm. Any encoder that emits symbols which cause the decoder to produce the original input is defined to be valid.

There are many possible strategies for the maximal-string matching that the encoder does, section 5.3.1 gives the code for one such algorithm. Regardless of which algorithm is used, and what tradeoffs are made between compression ratio and execution speed or space, the result can always be decoded by the simple decoder.

The compressed data consists of a mixture of unencoded literal characters and copy pointers which point to an earlier occurrence of the string to be encoded.

Compressed data contains two types of codewords:

LITERAL pass the literal directly to the uncompressed output.

COPY length, offset
go back offset characters in the output and copy length characters forward to the current position.

To distinguish between codewords, the copy length is used. A copy length of zero indicates that the following codeword is a literal codeword. A copy length greater than zero indicates that the

following codeword is a copy codeword.

To improve copy length encoding, a threshold value of 2 has been subtracted from the original copy length for copy codewords, because the minimum copy length is 3 in this compression scheme.

The maximum offset value is set at 32255. Larger offsets offer extremely low improvements in compression (less than 1 percent, typically).

No special encoding is done on the LITERAL characters. However, unary encoding is used for the copy length and copy offset values to improve compression. A start-step-stop unary code is used.

A (start, step, stop) unary code of the integers is defined as follows: The Nth codeword has N ones followed by a zero followed by a field of size START + (N * STEP). If the field width is equal to STOP then the preceding zero can be omitted. The integers are laid out sequentially through these codewords. For example, (0, 1, 4) would look like:

Codeword	Range
0	0
10x	1-2
110xx	3-6
1110xxx	7-14
1111xxxx	15-30

Following are the actual values used for copy length and copy offset:

The copy length is encoded with a (0, 1, 7) code leading to a maximum copy length of 256 by including the THRESHOLD value of 2.

Codeword	Range
0	0
10x	3-4
110xx	5-8
1110xxx	9-16
11110xxxx	17-32
111110xxxxx	33-64
1111110xxxxxx	65-128
1111111xxxxxxx	129-256

The copy offset is encoded with a (9, 1, 14) code leading to a maximum copy offset of 32255. Offset 0 is reserved as an end of compressed data flag.

Codeword	Range
0xxxxxxxxx	0-511
10xxxxxxxxxx	512-1535
110xxxxxxxxxx	1536-3583
1110xxxxxxxxxxx	3485-7679
11110xxxxxxxxxxxx	7680-15871
11111xxxxxxxxxxxxx	15872-32255

The 0 has been chosen to signal the start of the field for ease of encoding. (The bit generator can simply encode one more bit than is significant in the binary representation of the excess.)

The stop values are useful in the encoding to prevent out of range values for the lengths and offsets, as well as shortening some codes by one bit.

The worst case compression using this scheme is a 1/8 increase in size of the encoded data. (One zero bit followed by 8 character bits). After the character encoding, the worst case ratio is 3/2 to the original data.

The minimum copy length of 3 has been chosen because the worst case copy length and offset is 3 bits (3) and 19 bits (32255) for a total of 22 bits to encode a 3 character string (24 bits).

5.3 The Decoder

As mentioned previously, the compression is defined by the decoder. Any encoder that produced output that is correctly decoded is by definition correct.

The following is an implementation of the decoder, written more for clarity and as much portability as possible, rather than for maximum speed.

When optimized for a specific environment, it will run significantly faster.

```
 /* LZJU 90 Decoding program */

 /* Written By Robert Jung and Robert Ullmann, 1990 and 1991. */

 /* This code is NOT COPYRIGHT, not protected. It is in the true
    Public Domain. */

 #include <stdio.h>
 #include <string.h>
```

```
typedef unsigned char uchar;
typedef unsigned int  uint;

#define N          32255
#define THRESHOLD      3

#define STRTP          9
#define STEPP          1
#define STOPP         14
#define STRTL          0
#define STEPL          1
#define STOPL          7

static FILE *in;
static FILE *out;

static int   getbuf;
static int   getlen;
static long  in_count;
static long  out_count;
static long  crc;
static long  crctable[256];
static uchar xxcodes[] =
"+-0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ\
abcdefghijklmnopqrstuvwxyz";
static uchar ddcodes[256];

static uchar text[N];

#define CRCPOLY         0xEDB88320
#define CRC_MASK        0xFFFFFFFF
#define UPDATE_CRC(crc, c)  \
        crc = crctable[((uchar)(crc) ^ (uchar)(c)) & 0xFF] \
              ^ (crc >> 8)
#define START_RECD      "* LZJU90"



void MakeCrctable()     /* Initialize CRC-32 table */
{
uint i, j;
long r;
    for (i = 0; i <= 255; i++) {
        r = i;
        for (j = 8; j > 0; j--) {
            if (r & 1)
                r = (r >> 1) ^ CRCPOLY;
            else
```

```
                r >>= 1;
            }
        crctable[i] = r;
        }
}



int GetXX()             /* Get xxcode and translate */
{
int c;
    do {
        if ((c = fgetc(in)) == EOF)
            c = 0;
        } while (c == '\n');
    in_count++;
    return ddcodes[c];
}



int GetBit()            /* Get one bit from input buffer */
{
int c;
    while (getlen <= 0) {
        c = GetXX();
        getbuf |= c << (10-getlen);
        getlen += 6;
        }
    c = (getbuf & 0x8000) != 0;
    getbuf <<= 1;
    getbuf &= 0xFFFF;
    getlen--;
    return(c);
}



int GetBits(int len)        /* Get len bits */
{
int c;
    while (getlen <= 10) {
        c = GetXX();
        getbuf |= c << (10-getlen);
        getlen += 6;
        }
    if (getlen < len) {
        c = (uint)getbuf >> (16-len);
```

```
            getbuf = GetXX();
            c |= getbuf >> (6+getlen-len);
            getbuf <<= (10+len-getlen);
            getbuf &= 0xFFFF;
            getlen -= len - 6;
            }
        else {
            c = (uint)getbuf >> (16-len);
            getbuf <<= len;
            getbuf &= 0xFFFF;
            getlen -= len;
            }
        return(c);
    }



    int DecodePosition()    /* Decode offset position pointer */
    {
    int c;
    int width;
    int plus;
    int pwr;
        plus = 0;
        pwr = 1 << STRTP;
        for (width = STRTP; width < STOPP; width += STEPP) {
            c = GetBit();
            if (c == 0)
                break;
            plus += pwr;
            pwr <<= 1;
            }
        if (width != 0)
            c = GetBits(width);
        c += plus;
        return(c);
    }



    int DecodeLength()      /* Decode code length */
    {
    int c;
    int width;
    int plus;
    int pwr;
        plus = 0;
        pwr = 1 << STRTL;
```

```
        for (width = STRTL; width < STOPL; width += STEPL) {
            c = GetBit();
            if (c == 0)
                break;
            plus += pwr;
            pwr <<= 1;
            }
        if (width != 0)
            c = GetBits(width);
        c += plus;
    return(c);
    }


    void InitCodes()        /* Initialize decode table */
    {
    int i;
        for (i = 0; i < 256; i++) ddcodes[i] = 0;
        for (i = 0; i < 64; i++) ddcodes[xxcodes[i]] = i;
    return;
    }

    main(int ac, char **av)            /* main program */
    {
    int r;
    int j, k;
    int c;
    int pos;
    char buf[80];
    char name[3];
    long num, bytes;

        if (ac < 3) {
            fprintf(stderr, "usage: judecode in out\n");
            return(1);
            }

        in = fopen(av[1], "r");
        if (!in){
            fprintf(stderr, "Can't open %s\n", av[1]);
            return(1);
            }


        out = fopen(av[2], "wb");
        if (!out) {
            fprintf(stderr, "Can't open %s\n", av[2]);
            fclose(in);
```

```
        return(1);
            }

        while (1) {
            if (fgets(buf, sizeof(buf), in) == NULL) {
                fprintf(stderr, "Unexpected EOF\n");
            return(1);
                }
            if (strncmp(buf, START_RECD, strlen(START_RECD)) == 0)
                break;
            }

        in_count = 0;
        out_count = 0;
        getbuf = 0;
        getlen = 0;

        InitCodes();
        MakeCrctable();

        crc = CRC_MASK;
        r = 0;

        while (feof(in) == 0) {
            c = DecodeLength();
            if (c == 0) {
                c = GetBits(8);
                UPDATE_CRC(crc, c);
                out_count++;
                text[r] = c;
                fputc(c, out);
                if (++r >= N)
                    r = 0;
                }

            else {
                pos = DecodePosition();
                if (pos == 0)
                    break;
                pos--;
                j = c + THRESHOLD - 1;
                pos = r - pos - 1;
                if (pos < 0)
                    pos += N;
                for (k = 0; k < j; k++) {
                    c = text[pos];
                    text[r] = c;
                    UPDATE_CRC(crc, c);
```

```
                    out_count++;
                    fputc(c, out);
                    if (++r >= N)
                        r = 0;
                    if (++pos >= N)
                        pos = 0;
                    }
                }
            }

        fgetc(in); /* skip newline */

        if (fscanf(in, "* %ld %lX", &bytes, &num) != 2) {
            fprintf(stderr, "CRC record not found\n");
            return(1);
            }

        else if (crc != num) {
            fprintf(stderr,
                 "CRC error, expected %lX, found %lX\n",
                 crc, num);
            return(1);
            }

        else if (bytes != out_count) {
            fprintf(stderr,
                 "File size error, expected %lu, found %lu\n",
                 bytes, out_count);
        return(1);
            }

        else
            fprintf(stderr,
                 "File decoded to %lu bytes correctly\n",
                 out_count);

        fclose(in);
        fclose(out);
    return(0);
    }
```

5.3.1 An example of an Encoder

Many algorithms are possible for the encoder, with different tradeoffs between speed, size, and complexity. The following is a simple example program which is fairly efficient; more sophisticated implementations will run much faster, and in some cases produce

somewhat better compression.

This example also shows that the encoder need not use the entire window available. Not using the full window costs a small amount of compression, but can greatly increase the speed of some algorithms.

```
/* LZJU 90 Encoding program */

/* Written By Robert Jung and Robert Ullmann, 1990 and 1991. */

/* This code is NOT COPYRIGHT, not protected. It is in the true
   Public Domain. */

#include <stdio.h>

typedef unsigned char uchar;
typedef unsigned int  uint;

#define N          24000    /* Size of window buffer */
#define F            256   /* Size of look-ahead buffer */
#define THRESHOLD      3
#define K          16384    /* Size of hash table */

#define STRTP          9
#define STEPP          1
#define STOPP         14

#define STRTL          0
#define STEPL          1
#define STOPL          7

#define CHARSLINE     78

static FILE *in;
static FILE *out;

static int   putlen;
static int   putbuf;
static int   char_ct;
static long  in_count;
static long  out_count;
static long  crc;
static long  crctable[256];
static uchar xxcodes[] =
"+-0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ\
abcdefghijklmnopqrstuvwxyz";
uchar window_text[N + F + 1];
```

```
    /* text contains window, plus 1st F of window again
       (for comparisons) */

    uint hash_table[K];
    /* table of pointers into the text */

    #define CRCPOLY         0xEDB88320
    #define CRC_MASK        0xFFFFFFFF
    #define UPDATE_CRC(crc, c)  \
      crc = crctable[((uchar)(crc) ^ (uchar)(c)) & 0xFF] \
      ^ (crc >> 8)


    void MakeCrctable()     /* Initialize CRC-32 table */
    {
    uint i, j;
    long r;
        for (i = 0; i <= 255; i++) {
            r = i;
            for (j = 8; j > 0; j--) {
                if (r & 1)
                    r = (r >> 1) ^ CRCPOLY;
                else
                    r >>= 1;
            }
            crctable[i] = r;
        }
    }



    void PutXX(int c)           /* Translate and put xxcode */
    {
        c = xxcodes[c & 0x3F];
        if (++char_ct > CHARSLINE) {
            char_ct = 1;
            fputc('\n', out);
        }
        fputc(c, out);
        out_count++;
    }


    void PutBits(int c, int len)  /* Put rightmost "len" bits of "c" */
    {
        c <<= 16 - len;
        c &= 0xFFFF;
        putbuf |= (uint) c >> putlen;
```

```
        c <<= 16 - putlen;
        c &= 0xFFFF;
        putlen += len;
        while (putlen >= 6) {
            PutXX(putbuf >> 10);
            putlen -= 6;
            putbuf <<= 6;
            putbuf &= 0xFFFF;
            putbuf |= (uint) c >> 10;
            c = 0;
            }
    }


    void EncodePosition(int ch) /* Encode offset position pointer */
    {
    int width;
    int prefix;
    int pwr;
        pwr = 1 << STRTP;
        for (width = STRTP; ch >= pwr; width += STEPP, pwr <<= 1)
            ch -= pwr;
        if ((prefix = width - STRTP) != 0)
            PutBits(0xffff, prefix);
        if (width < STOPP)
            width++;
        /* else if (width > STOPP)
        abort(); do nothing */
        PutBits(ch, width);
    }


    void EncodeLength(int ch)   /* Encode code length */
    {
    int width;
    int prefix;
    int pwr;
        pwr = 1 << STRTL;
        for (width = STRTL; ch >= pwr; width += STEPL, pwr <<= 1)
            ch -= pwr;
        if ((prefix = width - STRTL) != 0)
            PutBits(0xffff, prefix);
        if (width < STOPL)
            width++;
        /* else if (width > STOPL)
        abort(); do nothing */
        PutBits(ch, width);
    }
```

```
    main(int ac, char **av)           /* main program */
    {
    uint r, s, i, c;
    uchar *p, *rp;
    int match_position;
    int match_length;
    int len;
    uint hash, h;

        if (ac < 3) {
            fprintf(stderr, "usage: juencode in out\n");
        return(1);
            }

        in = fopen(av[1], "rb");
        if (!in) {
            fprintf(stderr, "Can't open %s\n", av[1]);
        return(1);
            }

        out = fopen(av[2], "w");
        if (!out) {
            fprintf(stderr, "Can't open %s\n", av[2]);
            fclose(in);
        return(1);
            }

        char_ct = 0;
        in_count = 0;
        out_count = 0;
        putbuf = 0;
        putlen = 0;
        hash = 0;

        MakeCrctable();
        crc = CRC_MASK;

        fprintf(out, "* LZJU90 %s\n", av[1]);

        /* The hash table inititialization is somewhat arbitrary */
        for (i = 0; i < K; i++) hash_table[i] = i % N;

        r = 0;
        s = 0;

        /* Fill lookahead buffer */

        for (len = 0; len < F && (c = fgetc(in)) != EOF; len++) {
```

```
        UPDATE_CRC(crc, c);
    in_count++;
    window_text[s++] = c;
    }


    while (len > 0) {
    /* look for match in window at hash position */
    h = ((((window_text[r] << 5) ^ window_text[r+1])
            << 5) ^ window_text[r+2]);
    p = window_text + hash_table[h % K];
    rp = window_text + r;
    for (i = 0, match_length = 0; i < F; i++) {
            if (*p++ != *rp++) break;
            match_length++;
            }
    match_position = r - hash_table[h % K];
    if (match_position <= 0) match_position += N;

    if (match_position > N - F - 2) match_length = 0;
    if (match_position > in_count - len - 2)
        match_length = 0; /* ! :-) */

    if (match_length > len)
        match_length = len;
    if (match_length < THRESHOLD) {
        EncodeLength(0);
        PutBits(window_text[r], 8);
        match_length = 1;
        }
    else {
        EncodeLength(match_length - THRESHOLD + 1);
        EncodePosition(match_position);
        }

    for (i = 0; i < match_length &&
                    (c = fgetc(in)) != EOF; i++) {
            UPDATE_CRC(crc, c);
            in_count++;
        window_text[s] = c;
            if (s < F - 1)
            window_text
            [s + N] = c;
        if (++s > N - 1) s = 0;
        hash = ((hash << 5) ^ window_text[r]);
        if (r > 1) hash_table[hash % K] = r - 2;
        if (++r > N - 1) r = 0;
        }
```

```
        while (i++ < match_length) {
            if (++s > N - 1) s = 0;
            hash = ((hash << 5) ^ window_text[r]);
            if (r > 1) hash_table[hash % K] = r - 2;
            if (++r > N - 1 ) r = 0;
            len--;
                }
        }


        /* end compression indicator */
        EncodeLength(1);
        EncodePosition(0);
        PutBits(0, 7);

        fprintf(out, "\n* %lu %08lX\n", in_count, crc);
        fprintf(stderr, "Encoded %lu bytes to %lu symbols\n",
                in_count, out_count);

        fclose(in);
        fclose(out);

    return(0);
    }
```

5.3.2 Example LZJU90 Compressed Object

The following is an example of an LZJU90 compressed object. Using this as source for the program in section 5.3 will reveal what it is.

```
   Encoding: 7 LZJU90 Text

   * LZJU90 example
   8-mBtWA7WBVZ3dEBtnCNdU2WkE4owW+l4kkaApW+o4Ir0k33Ao4IE4kk
   bYtk1XY618NnCQl+OHQ61d+J8FZBVVCVdClZ2-LUI0v+I4EraItasHbG
   VVg7c8tdk2lCBtr3U86FZANVCdnAcUCNcAcbCMUCdicx0+u4wEETHcRM
   7tZ2-6Btr268-Eh3cUAlmBth2-IUo3As42laIE2Ao4Yq4G-cHHT-wCEU
   6tjBtnAci-I++
   * 190 081E2601
```

6. Alphabetical Listing of Defined Encodings

Keyword	Description	Section	Reference(s)
EDIFACT	EDIFACT format	3.5	
EDI-X12	EDI X12 format	3.5	ANSI X12
EVFU	FORTRAN format	3.4	
FS	File System format	3.6, 4	
Hex	Hex binary format	3.3	
LZJU90	LZJU90 format	3.7, 5	
LZW	LZW format	3.8	
Message	Encapsulated Message	3.2	STD 11, RFC 822
PEM, PEM-Clear	Privacy Enhanced Mail	3.10	RFC 1421-1424
PGP	Pretty Good Privacy	3.11	
Postscript	Postscript format	3.14	[8]
Shar	Shell Archive format	3.15	
Signature	Signature	3.12	
Tar	Tar format	3.13	
Text	Text	3.1	IS 10646
uuencode	uuencode format	3.9	
URL	external URL-reference	3.16	

7. Security Considerations

Security of content and the receiving (decoding) system is discussed in sections 3.10, 3.11, 3.15, and 4.2.10. The considerations mentioned also apply to other encodings and attributes with similar functions.

8. References

[1] Robinson, D. and R. Ullmann, "Encoding Header Field for Internet Messages", RFC 1154, Prime Computer, Inc., April 1990.

[2] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", STD 11, RFC 822, University of Delaware, August 1982.

[3] International Organization for Standardization, Information Technology -- Universal Coded Character Set (UCS). ISO/IEC 10646-1:1993, June 1993.

[4] Linn, J., "Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures" RFC 1421, IAB IRTF PSRG, IETF PEM WG, February 1993.

[5] Kent, S., "Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management", RFC 1422, IAB IRTF PSRG, IETF PEM, BBN, February 1993.

[6] Balenson, D., "Privacy Enhancement for Internet Electronic Mail: Part III: Algorithms, Modes, and Identifiers", RFC 1423, IAB IRTF PSRG, IETF PEM WG, TIS, February 1993.

[7] Kaliski, B., "Privacy Enhancement for Internet Electronic Mail: Part IV: Key Certification and Related Services", RFC 1424, RSR Laboratories, February 1993.

[8] Adobe Systems Inc., PostScript Language Reference Manual. 2nd Edition, 2nd Printing, January 1991.

[9] Rose, M. and E. Steffererud, "Proposed Standard for Message Encapsulation", RFC 934, Delaware and NMA, January 1985.

[10] Postel, J., "Simple Mail Transfer Protocol", STD 10, RFC 821, USC/Information Sciences Institute, August 1982.

[11] Borenstein, N., and N. Freed, "MIME (Multipurpose Internet Mail Extensions): Mechanisms for Specifying and Describing the Format of Internet Message Bodies", RFC 1341, Bellcore, Innosoft, June 1992.

[12] Borenstein, N., and M. Linimon, "Extension of MIME Content-Types to a New Medium", RFC 1437, 1 April 1993.

9. Acknowledgements

The authors would like to thank Robert Jung for his contributions to this work, in particular the public domain sample code for LZJU90.

10. Authors' Addresses

Albert K. Costanzo
AKC Consulting Inc.
P.O. Box 4031
Roselle Park, NJ 07204-0531

Phone: +1 908 298 9000
Email: AL@AKC.COM

David Robinson
Computervision Corporation
100 Crosby Drive
Bedford, MA 01730

Phone: +1 617 275 1800 x2774
Email: DRB@Relay.CV.COM

Robert Ullmann

Phone: +1 617 247 7959
Email: ariel@world.std.com

EXHIBIT B-6

ARTICLES

Pretty Good Privacy

July 1994 / Core Technologies / Pretty Good Privacy

This controversial security scheme for messages is a collection of international cryptographic methods

William Stallings

If you rely on E-mail for business or personal communications, beware: If you send messages over a network, they are subject to eavesdropping. And, as Oliver North and John Poindexter found out, if messages are stored in a file, they are subject to perusal months or even years later. You also need to be concerned about impersonation: That message asking leading questions may not be from your attorney at all, but from some hotshot reporter with excellent hacking skills.

What to do? PGP (Pretty Good Privacy) is the E-mail security package for Everyperson. Developed a few years ago by Phil Zimmermann, PGP combines confidential-ity and digital-signature capabilities to provide a powerful, virtually unbreakable, and easy-to-use package. Freeware versions are available for Windows, the Macintosh, DOS, OS/2, the Amiga, and other platforms.

Much has been written about the political and legal issues surrounding PGP. In this discussion I'll show how PGP works by explaining its four services for messages: authentication, confidentiality, compression, and E-mail compatibility. The figure "The PGP Process" shows the relationship among these four services.

Authentication

PGP employs the RSA public-key encryption scheme--a time-proven and easy-to-implement encryption method that is named after its inventors: Rivest, Shamir, and Adleman--and the MD5 (Message Digest version 5, also from Ron Rivest) one-way hash function to form a digital signature that assures the receiver that an incoming message is authentic (i.e., that it comes from the alleged sender and that it has not been altered). The sequence is as follows:

1. The sender creates a message.

2. MD5 is used to generate a 128-bit hash code of the message.

3. The hash code is encrypted with RSA using the sender's private key,

and the result is prepended to the message.

4. The receiver uses RSA with the sender's public key to decrypt and

recover the hash code.

5. The receiver generates a new hash code for the message and compares

it with the decrypted hash code. If the two match, the message is

accepted as authentic.

The combination of MD5 and RSA provides an effective digital-signature scheme. Because of RSA's strength, the receiver is assured that only the possessor of the matching private key can generate the signature. Because of MD5's strength, the receiver is assured that no one else can generate a new message that matches the hash code--and, hence, the signature--of the original message.

Each person's signature is independent and is therefore applied only to the document. Otherwise, signatures would have to be nested, with the second signer signing both the document and the first signature, and so on.

Confidentiality

Another basic service provided by PGP is confidential-ity, which is provided by encrypting messages to be transmitted or to be stored locally as files. In both cases, the conventional encryption algorithm known as IDEA (International Data Encryption Algorithm) is used. IDEA is a relatively new algorithm that is generally considered to be much stronger than the widely used DES algorithm.

In any conventional encryption system, the problem of key distribution must be addressed. In PGP, each conventional key is used only once. That is, a new key is generated as a random 128-bit number for each message. This session key is bound to the message and transmitted with it, as explained below.

1. The sender generates a message and a random 128-bit number to be used

as a session key for that message only.

2. The message is encrypted, using IDEA with the session key.

3. The session key is encrypted with RSA, using the receiver's public

key, and is prepended to the message.

4. The receiver uses RSA with his or her private key to decrypt and

recover the session key.

5. The session key is used to decrypt the message.

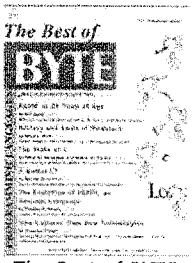
IDEA is substantially faster than RSA, so to reduce encryption time, the IDEA/RSA combination is used in preference to simply using RSA to directly encrypt the message. Also, the use of RSA solves the session-key distribution problem, because only the receiver is able to recover the session key that is bound to the message. Thus, to the extent that RSA is secure, the entire scheme is secure. To this end, PGP provides the user with several RSA key-size options:

-- Casual (384 bits): known to be breakable, but with much effort

-- Commercial (512 bits): possibly breakable by three-letter organizations

-- Military (1024 bits): generally believed to be unbreakable

Both confidentiality and authentication services can be used for the



The Best of BYTE
Volume 1:
Programming
Languages

In this issue of *Best of BYTE*, we bring together some of the leading programming language designers and implementors...

same message. First, a signature is generated for the plaintext message and is prepended to the message. Then, the plaintext message, along with the signature, is encrypted using IDEA, and the session key is encrypted using RSA.

In summary, when both services are used, the sender first signs the message with his or her own private key, then encrypts the message with a session key, and then encrypts the session key with the receiver's public key.

Compression

As a default, PGP compresses a message after applying the signature but before encryption. Compression has the benefit of reducing the size of an E-mail transmission.

Note in the figure "The PGP Process" that the signature is generated before compression. It is preferable to sign an uncompressed message so that you can store only the uncompressed message together with the signature for future verification. If you were to sign a compressed document, it would be necessary either to store a compressed version of th e message for later verification or to recompress the message when verification is required.

Message encryption is applied after compression to strengthen cryptographic security. Because the compressed message has less redundancy than the original plaintext, cryptanalysis is more difficult. The compression algorithm used for PGP is ZIP, a popular algorithm originally developed for DOS machines.

E-Mail Compatibility

When PGP is used, at least part of the block to be transmitted is encrypted. If only the signature service is used, then the message digest is encrypted (with the sender's private RSA key). If the confidentiality service is used, the message plus signature (if one is present) is encrypted (with a one-time IDEA key). Thus, all or part of the resulting block consists of a stream of arbitrary 8-bit octets.

However, many E-mail systems permit the use of only those blocks that consist of ASCII text. To accommodate this restriction, PGP provides the service of converting the raw 8 -bit binary stream to a stream of printable ASCII characters.

The scheme used for this purpose is known as radix-64 conversion. Each group of three octets of binary data is mapped into four ASCII characters. The use of radix-64 expands a message by 33 percent. Fortunately, the session key and signature portions of the message are relatively compact, and the plaintext message is compressed. In fact, the compression should be more than enough to compensate for the radix-64 expansion.

One noteworthy aspect of the radix-64 algorithm is that it blindly converts the input stream to radix-64 format regardless of content, even if the input happens to be ASCII text. Thus, if a message is signed but not encrypted, and the conversion is applied to the entire block, the output will be unreadable to the casual observer. This provides a certain level of confidentiality.

As an option, PGP can be configured to convert to radix-64 format only the signature portion of signed plaintext messages. This enable s the receiver to read the message without using PGP. PGP would still have to be employed to verify the signature, however.

Public-Key Management

As you can see, PGP contains a clever, efficient, interlocking set of functions and formats to provide an effective confidentiality and authentication service. But to complete the system, one final area needs to be addressed: public-key management.

In the PGP documentation, Phil Zimmermann neatly captures the importance of this area. "This whole business of protecting public keys from tampering is the single most difficult problem in practical public-key applications," he says. "It is the Achilles' heel of public-key cryptography, and a lot of software complexity is tied up in solving this one problem."

A number of approaches are possible within PGP for minimizing the risk of a user's public-key file containing false public keys, such as physically passing the key via surface mail or floppy disk, verifying a key by telephone, or transferring and confirming the key through a trusted third party. But another, more likely method is already being used: The use of a trusted key server and verification by monitoring the sender's PGP fingerprints in postings to Usenet newsgroups and other public forums.

PGP is young, strong, and coming on. It is already being widely used, and its growth is being fueled by the rapid growth in Internet use and the increasing reliance on E-mail for everything from legal documents to love letters. It is already the practice of many people to include their PGP fingerprint in E-mail messages. Expect to see more of this and to see such fingerprints appearing in print, as one does with this article, in the future.

Figure: The PGP Process

William Stallings is an independent consultant and a frequent contributor to BYTE. He is the author of over a dozen books on data communications and computer topics and is currently at work on a user's guide to PGP. This article is based on material fro m his most recent book, Network and Internetwork Security (Prentice-Hall, 1994). His PGP fingerprint is B1 4E 2A BD 96 08 8B A4 67 83 D1 09 FE 52 56 6C. You can contact him on the Internet at stallings@acm.org or on BIX c/o "editors."

Up Level Previous Search Comment Subscribe

EXHIBIT C

Performance and Accountability Report Fiscal Year 2007
Other Accompanying Information

Table of Contents | Management | Financial | Auditor | IG | Other

TABLE 13A: EX PARTE REEXAMINATION
(FY 2003 - FY 2007)

Activity	2003	2004	2005	2006	2007
Requests filed, total	**392**	**441**	**524**	**511**	**643**
By patent owner	136	166	166	129	124
By third party	239	268	358	382	519
Commissioner ordered	17	7	-	-	-
Determinations on requests, total [1]	**381**	**419**	**537**	**458**	**594**
Requests granted:					
By examiner	360	408	509	422	575
By petition	1	-	2	5	2
Requests denied	20	11	26	31	17
Requests known to have related litigation	**109**	**138**	**176**	**229**	**369**
Filings by discipline, total	**392**	**441**	**524**	**511**	**643**
Chemical	124	130	138	118	133
Electrical	118	156	188	228	275
Mechanical	150	155	198	165	235

Notes:
1: Past years' data have been revised from prior year reports. ***(back to text)***

< Previous Page | Next Page >

Is there a question about what the USPTO can or cannot do that you cannot find an answer for? Send questions about USPTO programs and services to the **USPTO Contact Center (UCC).** *You can suggest USPTO webpages or material you would like featured on this section by E-mail to the* **webmaster@uspto.gov**. *While we cannot promise to accommodate all requests, your suggestions will be considered and may lead to other improvements on the website.*

Last Modified: 12/21/2007 09:56:35

EXHIBIT A

WOODCOCK WASHBURN
INTELLECTUAL PROPERTY LAW
ATLANTA • PHILADELPHIA • SEATTLE

PHILADELPHIA OFFICE
CIRA CENTRE, 12TH FLOOR
2929 ARCH STREET
PHILADELPHIA, PA 19104-2891
215.568.3100
FAX: 215.568.3439

DAVID J. WOLFSOHN
215-564-2222
wolfsohn@woodcock.com

March 20, 2008

VIA EMAIL
John G. Day, Esquire
Ashby & Geddes
500 Delaware Avenue, 8th Floor
P.O. Box 1150
Wilmington, DE 19899

VIA EMAIL
Vineet Bhatia, Esquire
Susman Godfrey LLP
1000 Louisiana Street, Suite 5100
Houston TX 77002-5096

Re: Inovis, Inc. v. Classified Information, Inc. and Distance Digital Co., LLC
No. 07-459 (GMS)

Dear Counsel:

I write to inform you about our plans with respect to seeking an *ex parte* reexamination of U.S. Patent No. 5,812,669 ("the '669 Patent"). I inform you ahead of our filing because, as you may be aware, once the reexamination is requested, Inovis will be unable to stop the reexamination process from running its course and likely invalidating some or all of the claims of the '669 patent.

In case you are not familiar with the process, 35 U.S.C. §§301-307 of the Patent Act provide for *ex parte* reexamination of a patent. Section 302 provides that "[a]ny person at any time may file a request for reexamination by the Office of any claim of a patent on the basis of any prior art cited under the provisions of section 301 . . . ." We will within the next few weeks be filing such a request for reexamination with respect to the '669 Patent. As you may know, such requests are presently being granted by the Patent Office in 95% percent of requests, with the subsequent reexamination proceedings resulting in amended or cancelled claims over 70% of the time. Under the statute, the PTO is required to act on a request within 3 months of our filing the request.

Consequently, if any settlement discussions are to be had before the reexamination process is unstoppably commenced, they must be done now. We are still amenable to discussing resolution of this matter in accordance with the terms we previously have

WOODCOCK WASHBURN

proposed to you. If you are interested in engaging in such discussions, please give me a call. Otherwise, we will proceed as outlined above.

Very truly yours,

David J. Wolfsohn

DJW/mc

cc: (via email)
Julia Heaney, Esquire
Andrea E. Bates, Esquire
Lance D. Reich, Esquire
E. Michelle Tyde, Esquire
Erich M. Falke, Esquire
Jordan L. Jonas, Esquire

EXHIBIT B

DOCKET NO. INOV-0005

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent No: 5,812,669

Issued: September 22, 1998

Filed: July 19, 1995

Applicant: Lew Jenkins and Emmanuel K. Pasetes, Jr.

Title: METHOD AND SYSTEM FOR PROVIDING SECURE EDI OVER AN OPEN NETWORK

REQUEST FOR EX PARTE REEXAMINATION

MAIL STOP "*EX PARTE* REEXAMINATION"
ATTN: CENTRAL REEXAMINATION UNIT
COMMISSIONER FOR PATENTS
P.O. BOX 1450
ALEXANDRIA, VA 22313-1450

Sir:

Reexamination under 35 U.S.C. §§ 301-307 and 37 C.F.R. § 1.510 *et seq*. is requested of all claims of United States Patent No. 5, 812,669 ("the '669 Patent" or "the Patent." The '699 Patent issued on September 22, 1998, to Lew Jenkins and Emmanuel K. Pasetes, Jr. The filing date of the '669 Patent is July 19, 1995. The '669 Patent does not claim priority under 35 U.S.C. § 119 or § 120. This is a new reexamination request. The '669 Patent has not been previously reexamined.

The Requester is Inovis USA, Inc. The Requestor is aware of the following litigation involving the '669 Patent: Civil Action No. 07-459 (GMS) in the United States District Court for the District of Delaware.

Requester respectfully submits that substantial new questions regarding the patentability of claims 1-50 of the '669 Patent exist.

I. INTRODUCTION

The claims of the '669 Patent do not distinguish over the prior art and should not have been issued. The claims of the '669 Patent are directed to a system and method for conducting

Electronic Data Interchange (EDI) on an open network. As described in the opening paragraph of the specification of the '669 paragraph:

> The present invention relates to methods and systems for providing secure EDI over an open system network, such as the INTERNET, and particularly to an improved method and system for providing a secure EDI mailer over an open network which employs an RSA type public/private key encryption scheme in order to deliver secure authentication, and non-repudiation of both origin and receipt.

The '669 Patent summarizes how secure EDI is to occur, stating:

> The method and system of the present invention comprises using the AUTACK or EDI acknowledgement message as a document to provide the digital signature in a public/private key system in which the AUTACK is signed by an encrypted hash code from the EDI interchange communication which has been encrypted with the sender's private key, such as in an RSA type public/private key system, and is an improvement on such systems. Because the AUTACK or functional acknowledgement is sealed with the private key of the sender of the functional acknowledgement, the recipient of the original message, when the original sender decrypts the reply AUTACK message with the recipient's public key, he is assured that the intended recipient actually sent the reply AUTACK or acknowledgement and of the integrity of the receipt due to the correct hash code being detected.

The '669 Patent acknowledges that AUTACK is a "generic international standard of EDI.," *Id.*, col. 2, lines 15-17, and that "[a]s well known by users of EDI, an AUTACK message is a UN/EDIFACT standard for authentication and acknowledgement." *Id.*, col. 5, lines18-20.

II. THE PRIOR ART REFERENCES THAT PRESENT A SUBSTANTIAL NEW QUESTION OF PATENTABILITY

- U.S. Patent 5, 659,616 issued August 19, 1997, filed July, 16, 1996 as a continuation of an application filed July 19, 1994. (Sudia),
- United Nations TRADE/WF.4/R.1026/Add.2; "EDIFACT SECURITY IMPLEMENTATION GUIDELINES" (Feb. 22, 1994) (UN 1)
- United Nations TRADE/WF.4/R.1026/Add.3 "AUTACK: SECURE AUTHENTIFICATION AND ACKNOWLEDGEMENT MESSAGE" (Feb. 23, 1994) (UN 2)
- United Nations TRADE/WF.4/R.1026/Add.4 (Feb. 22, 1994) "MIG HANDBOOK UN/EDIFACT MESSAGE AUTACK" (UN 3)

- RFC 1505, "Encoding Header Field for Internet Messages," August 1993 (RFC), available at http://rfc.dotsrc.org/rfc/rfc1505.html
- BYTE, July 1994, "Pretty Good Privacy," available at http://www.byte.com/art/9407/sec12/art4.htm

III. STATEMENT POINTING OUT THE SUBSTANTIAL NEW QUESTION OF PATENTABILITY

The substantial new question of patentability is whether claims 1-50 of the '669 Patent are obvious under 35 U.S.C. § 103(a) on the basis of Sudia, UN 1, UN 2, UN 3, RFC, BYTE and the admissions of prior art set forth in the '669 Patent[1].

IV. CASE LAW RELEVANT TO THE OBVIOUSNESS ANALYSIS

"Section 103 forbids issuance of a patent when 'the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains.'" *KSR Int'l Co. v. Teleflex Inc.*, 127 S.Ct. 1727, 1734 (2007). The question of obviousness is resolved on the basis of underlying factual determinations including (1) the scope and content of the prior art, (2) any differences between the claimed subject matter and the prior art, (3) the level of skill in the art. *Graham v. John Deere Co.*, 383 U.S. 1, 17-18 (1966); See also *KSR*, 127 S.Ct. at 1734 ("While the sequence of these questions might be reordered in any particular case, the [Graham] factors continue to define the inquiry that controls.")

In *KSR*, the Supreme Court reaffirmed principles based on its precedent that "[t]he combination of familiar elements according to known methods is likely to be obvious when it does no more than yield predictable results." *Id.* The Court explained:

> When a work is available in one field of endeavor, design incentives and other market forces can prompt variations of it, either in the same field or a different one. If a person of ordinary skill can implement a predictable variation,

[1] It is proper to consider patentee's admissions in combination with patents and printed documents in a reexamination proceeding. *Ex parte McGaughey*, 6 USPQ2d 1334, 1337 (Bd. Pat. App. & Inter. 1988)

> §103 likely bars its patentability. For the same reason, if a technique has been used to improve one device, and a person of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill.

Id. at 1740. The operative question in this "functional approach" is thus "whether the improvement is more than the predictable use of prior art elements according to their established functions." *Id.*

When there is a design need or market pressure to solve a problem and there are a finite number of identified, predictable solutions, a person of ordinary skill has good reason to pursue the known options within his or her technical grasp. *Id.* If this leads to the anticipated success, it is likely the product not of innovation but of ordinary skill and common sense. *Id.* In that instance the fact that a combination was obvious to try might show that it was obvious under § 103. *Id.*

The Supreme Court's opinion in *United States v. Adams*, 383 U.S. 39, 40 (1966) is illustrative of the "functional approach" to be taken in cases where the claimed invention is a prior art structure altered by substituting one element in the structure for another known element. *KSR*, 127 S.Ct. at 1734. "The Court [in *Adams*] recognized that when a patent claims a structure already known in the prior art that is altered by the mere substitution of one element for another known in the field, the combination must do more than yield a predictable result. 383 U.S., at 50-51." *Id.*

The Federal Circuit has also recently incorporated the *KSR* decision stating "[a]n obviousness determine is not the result of a rigid formula disassociated from the consideration of the facts of a case. Indeed, the common sense of those skilled in the art demonstrates why some combinations would have been obvious where others would not." *Leapfrog Ent., Inc. v. Fisher-Price, Inc.*, 485 F.3d 1157, 1161 (Fed. Cir. 2007). The Federal Circuit relied in part on the fact that Leapfrog (the patent holder) had presented no evidence that the inclusion of a reader in the combined device was "uniquely challenging or difficult for one of ordinary skill in the art" or "represented an unobvious step over the prior art." *Id.* (citing *KSR*, 127 S.Ct. at 1740-41).

When the patentability of claims 1-50 of the '669 Patent is reexamined in light of the documents discussed below, it will be seen that the claimed system and method are obvious variations of EDI and data security systems that were well known prior to the filing date of the '669 Patent, especially when the principles of *KSR* are applied to

these facts. In the words of the Supreme Court in *KSR*, the claimed invention of the '669 Patent is, at best, no more than the predictable use of prior art elements according to their established functions.

V. DETAILED EXPLANATION OF THE SUBSTANTIAL NEW QUESTION OF PATENTABILITY

Claim 1 of the '669 Patent is written in Jepson form. As such, the contents of the preamble are admittedly prior art. *In re Fout,* 675 F.2d 297, 301 (CCPA 1982); 37 CFR 1. 75(e). The admissions made in regard to claim 1 of course carry over to independent method claim 35. In allowing the claims of the '669 Patent the Examiner indicated in his Statement of Reasons for Allowance that the prior art of record would not anticipate or tend to render obvious the claimed "improvement" in claim 1 or the method of claim 35, noting the use of the "associated EDI acknowledgement message." Paper No. 12, page 2. As developed below, the subject matter of each claim as a whole including claim 1, preamble and "improvement," is obvious within the meaning of the statute.

By way of background, there are several different standards for EDI. One is the United Nations EDIFACT standard (UN/EDIFACT). EDI was originally conducted on value added networks (VANs). EDI began its migration to the internet prior to the July 19, 1995, filing date of the '669 Patent. .For example, RFC describes multi-part, multi-structured internet messages that can be used in EDI including EDIFACT. See, e.g., pages 1 and 7. Security is needed for EDI messages sent over a network, especially an open network such as the internet. See, e.g., RFC, page 8 (non-encrypted EDIFACT can provide non-repudiation but not confidentiality.). U. S. Patent No. 5,659,616 (Sudia) provides background information on digital signatures and encrypting electronic messages used in commercial electronic documents.

Sudia illustrates in Fig. 1a the well known symmetric encryption system. Sudia illustrates in Fig. 1b the well known asymmetric encryption method. Sudia illustrates in Fig. 2 a prior art process of using an asymmetric key to provide a digital signature. As seen, the sender digests the message with a hash function then encrypts the message with the sender's private key. The result is appended to the message as its signature. The recipient decrypts the message using the sender's public key and hashes the message using the same hash function. If the decrypted hash is the same as the hash accompanying the message, the signature is verified. As

acknowledged by the '669 Patent, RSA type public key/private key encryption systems were well known at the time of the '669 Patent invention as well as hashes such as MD5. See, e.g., '669 Patent, col. 1, lines 13-42; col. 4, lines 48-56; col. 5, lines 11-17.

In addition to Sudia, BYTE establishes that it was well known prior to the filing date of the '669 Patent to use PGP security as email security. As explained RSA public key/private key and MD5 are used as a digital signature. BYTE also describes message encryption by way of IDEA instead of DES and digitally signing the encrypted message with RSA security. Note that RFC discusses PGP and EDI in regard to internet email at, e.g., pages 8-9.

It does not appear that any documents in the file of the '669 Patent describe the admittedly known and conventional AUTACK message as used in UN/EDIFACT. Prior art documents identified above as UN 1, UN 2 and UN 3 fill this vacuum and describe EDI security in UN/EDIFACT, especially the use of the AUTACK message.

UN 1 describes EDIFACT SECURITY IMPLEMENTATION GUIDELINES. UN 1 is specific to the EDI interchange message itself and how to secure the EDI interchange message. UN 1 states that the security functions applied to the EDI interchange message provide non-repudiation of origination (NRO), message origin authentication (AUT) and the integrity of the message (INT). UN 1, page 8. Among the choices of security techniques described by UN 1 are asymmetric algorithms and hashing functions. UN 1, page 33, UN 1 describes the combine use of asymmetric algorithms and hashing functions as providing NRO and AUT as well as the use of asymmetric algorithms for INT. UN 1,page 34.

UN 1 describes at page 31 the use of a bilateral agreement between the business partners in which security services, algorithms, codes key management methods etc are agreed upon. The bilateral agreement may also involve a third party acting as certification authority. UN 1, page 31. RSA public key/private key security and MD 5 hashes are used in securing the EDIFACT interchange. UN 1, page 25.

In describing how to protect an EDIFACT message at pages 30, *et seq*, UN 1 states:

> Non-repudiation of receipt is a service to be initiated by the receiver. It could either be requested by the sender in the message or mandated in an interchange agreement. A special message, AUTACK, is being developed to convey the receipt.

UN 2 and UN 3 describe the AUTACK message and the security services used with that document. The AUTACK message is described by UN 2, page 4 as:

> A secure authentication and acknowledgement message used as an authentication message is sent by the originator of messages and interchanges sent separately or by a party having authority to act on behalf of the originator, to facilitate message origin authentication, validation of integrity of content, validation of message sequence integrity or non-repudiation of origin of these messages.
>
> A secure authentication and acknowledgement message used as an acknowledgement message is sent by the recipient of previously received messages or by a party having authority to act on behalf of the recipient, to facilitate confirmation of receipt, validation of integrity of content, validation of completeness or non-repudiation of receipt of these messages.
>
> This AUTACK message can apply to one or more messages from one or more interchanges or to one or more interchanges.
>
> The security services are provided by cryptographic mechanisms applied to the content of the original messages or interchanges. The results of these mechanisms form the body of the AUTACK message, supplemented by the relevant data such as the reference to the cryptographic methods used, the reference number and the date and time of the original entities.
>
> The AUTACK message is completed by the standard security header and trailer groups.

UN 2 states at page 5 that "at least one security method should be applied to AUTACK…"

UN 3 also describes the AUTACK message as does UN 2, see UN 3, page 5 and also describes at pages 30-32 that security methods used include public key/private key systems involving hashes ,e.g., RSA and MD5.

Taking a step back it is seen that each element of the system of claim 1, both in the admittedly old preamble and the allegedly new "improvement," were well known for the exact same function they are used in claim1. Thus, it is seen that claim 1 is only a combination of prior art elements assembled as instructed in the prior art. Under the principles set forth in *KSR*, the subject matter of each of claims 1-50 of the '669 Patent is obvious.

a. Claim 1

Patentees admit in the preamble of claim 1 that the following elements are known[2]:

- public key/private key secure communication system selectively interconnecting a plurality of computers over an open public network (RSA security, Sudia, Byte, UN 1, UN 2, UN 3, RFC)

[2] Parentheticals show where the admitted elements are documented in the prior art cited above. See also the claim chart set forth below.

- sender and receiver computers (all references)
- sender and receiver computers exchanging secure digital messages (Sudia, BYTE UN 1, UN 2, UN 3)
- sender computer is associated with first public key and first private key (RSA security, Sudia, Byte, UN 1, UN 2, UN 3)
- recipient computer is associated with second public key and second private key (RSA security, Sudia, Byte, UN 1, UN 2, UN 3)
- digital messages comprising EDI interchange communications between the sender and receiver computers(UN 1, UN 2, UN 3,RFC)
- the EDI interchange communication having an associated acknowledgement message (UN 2, UN 3).

The "improvement" of claim 1 first involves encrypting the EDI interchange communication by a hash and inserting the hash in a predetermined location in the associated EDI acknowledgement message. This step is encompassed by the EDIFACT AUTACK mechanism. When used as non-repudiation of origin, AUTACK is described:

> 1.3.1.1 Hashing of references and secured AUTACK
> The secured entity (message or interchange) is referenced in an occurrence of the USX segment. Underneath it, at least one occurrence of USY is required referring to a header group of the present AUTACK thanks to the use of a "0534 Security result link" data-element. This particular header only contains the description of a hashing method, which is the one applied to the referred entity from its beginning service segment up to and including its ending service segment end. In the applicable header, the data-element "501 Security function; coded" refers to "referenced entity integrity". In this case, the AUTACK message must be secured (signed) by use of at least one other header and its corresponding trailer. It is advised that the header applicable for AUTACK security provides for encompassing security, encompassing the header used to indicate the referenced entity hashing method.

UN 2, pages 2-3. Thus, AUTACK is used with a hashed, encrypted EDI message. See also UN 2, page 3 ("The security services are provided by cryptographic mechanisms applied to the content of the original messages or interchanges. The results of these mechanisms form the body of the AUTACK message, supplemented by the relevant data such as the reference to the

cryptographic methods used, the reference number and the date and time of the original entities.")[3]

Next the "improvement" of claim 1 includes means for computing a second hash of the associated EDI acknowledgement message and digitally signing the associated EDI acknowledgement message with the sender's private key. This part of the "improvement" is only the use of a known security mechanism (RSA and MD5) to digitally sign the EDI acknowledgement message. AUTACK is to be secured. See, e.g., UN 3, page 4 ("The security functions NRO [non-repudiation of origin], AUT [message origin authentication] and INT [integrity] are applied to the AUTACK itself.").

The second hash is inserted in a predetermined location in the EDI acknowledgement message. UN 3 states at pages 11-12 that the SECURITY HEADER segment specifies the security service applied to the AUTACK message or the referenced message or interchange. Thus, AUTACK would contain in an appropriate place information concerning the second hash.

The "improvement" next requires means for transmitting the EDI interchange communication along with the digitally signed associated EDI acknowledgement message to the recipient computer. This aspect of the "improvement" is clearly taught by the UN documents involving EDI as the secured AUTACK and associated EDI interchange will be sent to the recipient computer. As set forth above, EDI was conducted over open public networks such as the internet (RFC) as opposed to VANs well prior to the filing date of the '669 patent.

The "improvement" next requires means for the recipient computer to process the received EDI interchange communication and the digitally signed EDI acknowledgement message for providing authentication and non-repudiation of the EDI interchange communication received from the sender computer. As discussed above, one of the purposes of the AUTACK is to provided authenticity and non repudiation of origin. This is accomplished in the "improvement" by providing means to decrypt the second hash which is of the EDI acknowledgement message with the sender's public key.. This is only the well known function of decrypting of the RSA, MD5 digital signature. See, e.g. UN 1, pages 36-49 (examples use hashes and RSA to secure EDIFACT message).

[3] As discussed above, UN 1 describes how the EDIFACT interchange message is secured using convention RSA and MD5.

The "improvement" concludes with a "whereby" clause which specifies that secure private EDI interchange communications can occur over said open public network while providing authentication and non-repudiation of the EDI communications. Again, this is one of the purposes of EDIFACT security and secure AUTACK.

b. Claim 2

Claim 2 provides for means that computes a third hash of the received EDI acknowledgement message and compares the third hash with the decrypted second hash. If the third and second hashes match, integrity of the of the EDI acknowledgement message as well as non-repudiation of origin is provided. As set forth above, one of the purposes of AUTACK is to provide for the integrity and non-repudiation of origin. See UN 2, page 2, UN 3, page 5. Comparing the decrypted second hash with a third hash of the received EDI acknowledgement message is the conventional manner in which digital signatures based upon RSA and MD5 are used. See Sudia and BYTE.

c. Claim 3

Claim 3 requires means for computing a fourth hash of the received EDI interchange communication and comparing the computed hash with the first hash of the RDI interchange communication. If the first and fourth hash match, the integrity and non-repudiation of origin of the EDI interchange message is provided. Since the EDI interchange message was hashed as a security feature by the sender computer, it is obvious to compute the fourth hash and compare it with the first hash to determine integrity and non-repudiation of origin of the EDI interchange communication. See Sudia and BYTE.

d. Claim 4

Claims 4 requires means for creating a reply EDI acknowledgement message and transmitting the message to the sender computer over the open public network. As taught by UN 2, page 4 and UN 3, page 5, AUTACK can be used by the recipient confirm receipt, validation of integrity of content, validation of completeness or non-repudiation of receipt of a received message. Claim 4 also requires means for computing a fifth hash of the reply EDI acknowledgement message and digitally signing the fifth hash by encrypting it with recipient's private key. and inserting the digitally signed hash into a predetermined position of the transmitted reply EDI acknowledgement message. This is only the use of a well known prior art digital signature, RSA and MD5, with a communication, reply AUTACK, that is to be secured.

e. Claim 5

Claim 5 requires means associated with the sender's computer for receiving the transmitted reply EDI acknowledgement message, decrypting the fifth hash with the recipient's computer's public key, computing a sixth hash of the received reply acknowledgement EDI message, comparing the sixth hash against the decrypted fifth hash to provide an indication of integrity of the received reply EDI acknowledgement message and non-repudiation of the reply EDI acknowledgement message so that non-repudiation of receipt the EDI interchange communication is established by the sender computer.

These means are directly taught or suggested by the normal use of a secure AUTACK as an "acknowledgement message…sent by the recipient of previously received [EDIFACT EDI] messages…to facilitate confirmation of receipt, validation of integrity of content, validation of completeness or non-repudiation of receipt of these messages." UN 2, page 4, UN 3, page 5. One would need to compute the specified sixth hash and use the recipient's public key in order to decrypt the reply AUTACK.

f. Claims 6 and 12

Claims 6 and 12, while of varying dependency, require that the means for creating the reply EDI acknowledgement message also comprise means for inserting the fourth hash in a predetermined location in the transmitted reply EDI acknowledgement message, means associated with the sender computer for comparing the fourth hash with the first hash to provide an indication of integrity and authenticity of the EDI interchange when the first and fourth hash match.

As explained above in regard to claim 3, since the EDI interchange message was hashed as a security feature by the sender computer, it is obvious to compute the fourth hash and compare it with the first hash to determine integrity and non-repudiation of origin of the EDI interchange communication. To do so in the context of the reply EDI acknowledgement message would also have been obvious. as taught by UN 2 and UN 3.

g. Claim 14

Claim 14 depends from claim 1 and begins to define another embodiment. Claim 14 requires means associated with the recipient computer for creating a reply EDI acknowledgement message and transmitting the message to the sender computer over the open public network. The reply EDI acknowledgement message creating means computes a third hash of the reply EDI

acknowledgement message and digitally signs the message by encrypting the hash with the recipient's private computer key and inserting the digitally signed hash into a predetermined location in the transmitted reply EDI acknowledgement message.

This claim is simply the application of RSA, MD5 security to the reply EDI acknowledgement message. As set forth above, AUTACK will contain the hash in an appropriate location.

h. Claim 16

Claim 16 requires means associated with the sender computer for receiving the transmitted reply EDI acknowledgement message and decrypting the encrypted third hash with the recipient computer's public key to verify the digital signature. In addition, the means is to compute a fourth hash of the received reply EDI acknowledgement message and compare the fourth hash against the decrypted third hash to provide an indication of integrity and non-repudiation of origin of the reply EDI acknowledgement message which in turn provides non-repudiation of receipt of the EDI interchange communication.

These means are directly taught or suggested by the normal use of a secure AUTACK used as an "acknowledgement message…sent by the recipient of previously received [EDIFACT EDI] messages…to facilitate confirmation of receipt, validation of integrity of content, validation of completeness or non-repudiation of receipt of these messages." UN 2, page 4, UN 3, page 5. One would need to compute the specified fourth hash and use the recipient's public key in order to decrypt the reply AUTACK.

i. Claim 17

Claim 17 appears to be in essence a duplicate of claim 16, differing in that claim 17 does not require the open public network to be the internet. The same analysis used in regard to claim 16 applies to claim 17.

j. Claims 7-11, 13, 15, 18-30, and 32-34

These claims require the use of AUTACK, MD5 hash, RSA type cryptographic communication system and/or the internet. As explained above, each of these features were well known to be used in EDI at the time of the invention of the '669 Patent **k. Claim 31**

Claim 31 requires means generating a trading partner agreement communication between the sender and recipient computers that contains the public keys of the trading partners. This taught at UN 1, page 31.

k. Claims 35-50

Independent method claim 35 is not in Jepson form but is quite similar to independent system claim 1. The admissions made in regard to the preamble of claim 1 apply equally to claim 35. The analysis of claim 1 applies to claim 35.

Claims 37, 38 and 48, while of varying dependency, are directed to the use of a digitally signed AUTACK as acknowledgement of receipt to provide non-repudiation of receipt of the EDI interchange communication. These steps are directly taught or suggested by the normal use of a secure AUTACK as an "acknowledgement message…sent by the recipient of previously received [EDIFACT EDI] messages…to facilitate confirmation of receipt, validation of integrity of content, validation of completeness or non-repudiation of receipt of these messages." UN 2, page 4, UN 3, page 5.

As to claims 39 and 43, the use of secured AUTACK with an EDI interchange communication also provides, at the recipient computer, non-repudiation of origin. UN 2, page 4, UN 3, page 5.

Claims 36, 40-42, 44-47, 49 and 50 are directed to the use of AUTACK, the internet and/or a RSA type cryptographic communication system. As explained above, these are well known and obvious to use in EDI.

DOCKET NO. INOV-0005

CLAIM CHART FOR '669 PATENT

Claim 1

	Admitted Prior Art	**UN docs**	**RFC**	**BYTE**	**Sudia**
In a public key/private key secure communication system for selectively interconnecting a plurality of computers over an open public network, said plurality of computers comprising a sender computer and a recipient computer, said sender and recipient computers exchanging secure digital messages there between, said sender computer having a first associated public key and a first associated private key, said recipient computer having a second associated public key and a second associated private key, said digital messages	Claim 1 is in Jepson format. As such the limitations in the preamble are admitted prior art. *In re Fout,* 675 F.2d 297, 301 (CCPA 1982); 37 CFR 1. 75(e). Patentees admit that using RSA public/private key encryption for providing secure messages over the internet was known. Col. 1, lines 13-23; col. 4, lines 48-56. EDI is admittedly old	Each of UN 1, UN 2 and UN	Internet email useful for EDI, need for security. See p. 8	BYTE describes Pretty Good Privacy (PGP) system for email security. PGP uses RSA (public key/private key)	Sudia describes various schemes for encrypting electronic documents including using symmetric encryption (Fig. 1a), asymmetric encryption (Fig. 1b) and hashes with public key/private key (Fig. 2).

comprising an EDI interchange communication between said sender computer and said recipient computer, said EDI interchange communication having an associated EDI acknowledgment message; the improvement in said secure open network communication system comprising	AUTACK known EDI acknowledgement message. Col. 5, lines18-20.	3 describe EDI. UN 1, page 8 describes the use of security schemes applied to the message including integrity (INT) and non-repudiation of origin (NRO). AUTACK can be used by sender as authentication message and by the recipient as an acknowledgement message. UN 2, page 4; UN 3, page 5 UN 2, page 6 in discussing Segment Group 1: USH-USA-SG2 states "A group of segments providing all security information necessary for the integrity, authentication, and non-repudiation of origin or of			

		receipt of all entities referenced in the AUTACK message and the AUTACK message itself. EDI messages including AUTACK may be secured using hashes, e.g., MD5, and public key/private key security system, e.g., RSA. UN 1, page 15, (code 6 MD5 and RSA).			
means for computing a first hash for said EDI interchange communication from said sender computer	MD5 hash is conventional. Col. 5, lines 11-16.	EDI interchange message is encrypted to assure integrity using hash. UN 1, pages 33-34, (hashing function and asymmetric algorithm (RSA) can be used for message content integrity)		MD5 calculated for email message	Hash used as part of encryption system, e.g., Fig. 2.
means for inserting said first hash in a predetermined		Results of cryptographic mechanisms applied to		Encrypted hash prepended to email.	Encrypted hash appended to electronic

location in said associated EDI acknowledgment message		content of original message are part of the body of AUTACK. UN 2, page 4. See also UN 3, page 11 ("The SECURITY HEADER) segment specifies the security service applied to the AUTACK message or the referenced message or interchange.")			message. Col. 2, lines 47-64.
means for computing a second hash of said associated EDI acknowledgment message	MD5 hash is conventional. Col. 5, lines 11-16.	Hash is used as part of secure digital signature. UN 1, pages 33-34 (hashing function and asymmetric algorithm (RSA) can be used for NRO) UN 2, page 5, ("at least one security method should be applied to AUTACK…)		MD5 is calculated for email message	Hash used as part of encryption system, e.g., Fig. 2.
means for digitally signing said associated	Patentees admit that using RSA public/private	Asymmetric algorithm (RSA public		PGP uses RSA—sender	Public key/private key security

EDI acknowledgment message, said message digitally signing means comprising means for encrypting said second hash with said sender computer's private key	key encryption for providing secure messages over the internet was known. Col. 1, lines 13-23; col. 4, lines 48-56.	key/private key) can be used with hashing function. UN 3, page 32..		signs with private key	for electronic messages known. (Fig. 2).
means for inserting said second hash in a predetermined location in said associated EDI acknowledgment message		Results of cryptographic mechanisms applied to content of original message are part of the body of AUTACK. UN 2, page 4 See also UN 3, page 11 ("The SECURITY HEADER) segment specifies the security service applied to the AUTACK message or the referenced message or interchange.")		Encrypted hash prepended to email.	Encrypted hash appended to electronic message. Col. 2, lines 47-64.
means for transmitting said EDI interchange communication along with said digitally signed associated EDI	EDI on internet is admitted to be known.		Internet email useful for EDI, need for security. P. 8	Encrypted email is sent over internet.	

acknowledgment message to said recipient computer over said open public network; and					
means associated with said recipient computer for receiving and processing said received EDI interchange communication and said digitally signed EDI acknowledgment message for providing authentication and non-repudiation of said EDI interchange communication from said sender computer, said means comprising means for decrypting said encrypted second hash with said sender computer's public key; whereby secure private EDI interchange communications can occur over said open public network while providing authentication	Decrypting MD5 hash is known; RSA (public key/private key) is known.	Use of sender's public key to decrypt RSA encoded EDI messages is well known. See, e.g., UN 1, pages 39-43, Example 2. UN 2, page 4 and UN 3, page 5—Use of AUTACK by sender provides for validation of integrity of content and non-repudiation of origin.		PGP uses RSA where recipient uses sender's public key to decrypt message.	

and non-repudiation of said EDI communications.					

Claim 2

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 1 wherein said means associated with said recipient computer further comprises means for computing a third hash of said received EDI acknowledgement message; and means for comparing said third hash with said decrypted second hash from said received EDI acknowledgement message, said comparing means comprising means for providing an indication of integrity of said EDI acknowledgement message and non-repudiation of origin when said decrypted second hash and said third	Computing MD5 is known.	UN 1, pages 33-34 describes the conjoint use of hash algorithm and RSA to secure EDI messages. As explained by BYTE this means that a hash value based upon the received message will be calculated and compared with the decrypted hash value.	.	PGP includes calculating a hash value based upon the received message and comparing it with the decrypted hash value.	Hash used as part of encryption system, e.g., Fig. 2.

hash match					

Claim 3

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 2 wherein said means associated with said recipient computer further comprises means for computing a fourth hash of said received EDI interchange communication; and means for comparing said fourth hash of said received EDI interchange communication with said first hash in said received EDI acknowledgement message, said comparing means comprising means for providing an indication of integrity and verification of authenticity of said EDI interchange communication and non-repudiation of		When the EDI message is encrypted by using a hash value (first hash), a fourth hash will need to calculated to compare with the decrypted first hash to verify the content and provide non-repudiation of origin of the EDI message.		PGP includes calculating a hash based upon the received message and comparing it with the decrypted hash.	See Fig. 2 for comparing hashes.

origin when said first and fourth hash match.					

Claim 4

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 3 wherein said means associated with said recipient computer further comprises means for creating a reply EDI acknowledgement message and transmitting said reply EDI acknowledgement message to said sender computer over said open public network, said reply EDI acknowledgement message creating means comprising means for computing a fifth hash of said reply EDI acknowledgement message and for digitally signing said fifth hash by encrypting said fifth hash with said recipient		AUTACK is used as acknowledgement message of receipt by the recipient of previously received messages. UN 2, page 4; UN 3, page 5 EDI messages including AUTACK are to be secured. Conventional security systems using hashes, e.g., MD5, and public key/private key security system, e.g., RSA, are used in EDI. UN 1, page 15, (code 6 MD5 and RSA).		PGP includes calculating a hash based upon the received message and comparing it with the decrypted hash.	See Fig. 2.

computer's private key; and means for inserting said digitally signed fifth hash into a predetermined location in said transmitted reply EDI acknowledgement message.					

Claim 5

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 4 further comprising means associated with said sender computer for receiving said transmitted reply EDI acknowledgement message, and for decrypting said encrypted fifth hash with said recipient computer's public key for verifying said digital signature of said reply EDI acknowledgement message; and means for computing a sixth hash of said		UN 2, page 4 and UN 3, page 5 state that a secure AUTACK sent from the recipient to the sender is non-repudiation of receipt.		PGP includes calculating a hash based upon the received message and comparing it with the decrypted hash.	See Fig. 2.

received reply reply [sic] EDI acknowledgement message; and means for comparing said sixth hash against said decrypted fifth hash, said comparing means comprising means for providing an indication of integrity of said received reply EDI acknowledgement message and non-repudiation of origin of said reply EDI acknowledgement message; whereby non-repudiation of receipt of said EDI interchange communication is established by said sender computer.					

Claim 6

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 5 wherein said means for creating said reply EDI acknowledgement message further		UN 3, page 5 states that a secure AUTACK sent from the recipient to the sender is validation of integrity of content.		PGP includes calculating a hash based upon the received message and comparing it with the decrypted	See Fig. 2.

comprises means for inserting said fourth hash in a predetermined location in said transmitted reply EDI acknowledgement message, and said means associated with said sender computer further comprises means for comparing said fourth hash in said received reply EDI acknowledgement message with said first hash, said comparing means providing an indication of integrity and authenticity of said EDI interchange when said first and fourth hash match.				hash.	

Claim 7

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 6 wherein said EDI acknowledgement message comprises an AUTACK message.	AUTACK is a known EDI acknowledgement message. Col. 5, lines18-20.	UN 2, page 4, UN 3, page 5			

DOCKET NO. INOV-0005

Claim 8

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 7 wherein said reply EDI acknowledgement message comprises an AUTACK message.	AUTACK is a known EDI acknowledgement message. Col. 5, lines18-20.	UN 2, page 4; UN 3, page 5			

Claim 9

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 8 wherein each of said hashes comprise an MD5.	MD5 hash is conventional. Col. 5, lines 11-16.	MD5 is a known hash algorithm in EDI. UN 3, page 21 (MD5 listed as hash algorithm)		MD5 hash used in PGP.	

Claim 10

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 9 wherein said public and private keys	Patentees admit that using RSA public/private key encryption for providing secure messages over the internet was known. Col. 1,	UN 3, page 32—RSA security used in EDI AUTACK.		PGP uses RSA type security	Fig. 2 illustrates RSA type security

comprise an RSA type cryptographic communication system.	lines 13-23; col. 4, lines 48-56.				

Claim 11

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 10 wherein said open public network comprises the Internet.	EDI on internet is admitted to be known		Internet email is used for EDI, security needed. See, p. 8.	PGP used for internet email	

Claim 12

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 4 wherein said means for creating said reply EDI acknowledgement message further comprises means for inserting said fourth hash in a predetermined location in said transmitted reply EDI acknowledgement		Results of cryptographic mechanisms applied to content of original message are part of the body of AUTACK. UN 2, page 4. E.g., Security header, USH. UN 1, page 6. See also UN 3, page 11 ("The SECURITY HEADER) segment		PGP provides for comparing hash values for confirming authenticity and integrity.	See Fig. 2

message, and said means associated with said sender computer further comprises means for comparing said fourth hash in said received reply EDI acknowledgement message with said first hash, said comparing means providing an indication of integrity and authenticity of said EDI interchange when said first and fourth hash match.		specifies the security service applied to the AUTACK message or the referenced message or interchange.")			

Claim 13

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 1 wherein said open public network comprises the Internet.	EDI on internet is admitted to be known		Internet email is used for EDI, security needed.. P. 8	PGP used in internet email.	

DOCKET NO. INOV-0005

Claim 14

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 1 wherein said means associated with said recipient computer further comprises means for creating a reply EDI acknowledgement message and transmitting said reply EDI acknowledgement message to said sender computer over said open public network, said reply EDI acknowledgement message creating means comprising means for computing a third hash of said reply EDI acknowledgement message and for digitally signing said third hash by encrypting said third hash with said recipient computer's private key; and		UN 2, page 4 and UN 3, page 5 state that a secure AUTACK sent from the recipient to the sender is validation of integrity of content. UN 3, page 32—RSA security used in EDI AUTACK. UN 3, page 11		PGP includes calculating a hash based upon the received message and comparing it with the decrypted hash	See Fig. 2.

means for inserting said digitally signed third hash into a predetermined location in said transmitted reply EDI acknowledgement message.		("The SECURITY HEADER) segment specifies the security service applied to the AUTACK message or the referenced message or interchange.")			

Claim 15

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 14 wherein said open public network comprises the Internet.	EDI on internet is admitted to be known.		Internet email is used for EDI, security needed. See p. 8	PGP used for internet email.	

DOCKET NO. INOV-0005

Claim 16

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 15 further comprising means associated with said sender computer for receiving said transmitted reply EDI acknowledgement message, and for decrypting said encrypted third hash with said recipient computer's public key for verifying said digital signature of said reply EDI acknowledgement message; and means for computing a fourth hash of said received reply reply [sic] EDI acknowledgement message; and means for comparing said fourth hash against said decrypted third hash, said comparing means		UN 2, page 4 and UN 3, page 5 state that a secure AUTACK sent from the recipient to the sender is non-repudiation of receipt.		PGP includes calculating a hash based upon the received message and comparing it with the decrypted hash	See Fig. 2.

comprising means for providing an indication of integrity of said received reply EDI acknowledgement message and non-repudiation of origin of said reply EDI acknowledgement message; whereby non-repudiation of receipt of said EDI interchange communication is established by said sender computer.					

Claim 17

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 14 further comprising means associated with said sender computer for receiving said transmitted reply EDI acknowledgement message, and for decrypting said encrypted third hash with said		UN 2, page 4 and UN 3, page 5 state that a secure AUTACK sent from the recipient to the sender is non-repudiation of receipt		PGP includes calculating a hash based upon the received message and comparing it with the decrypted hash	See Fig. 2

recipient computer's public key for verifying said digital signature of said reply EDI acknowledgement message; and means for computing a fourth hash of said received reply reply [sic] EDI acknowledgement message; and means for comparing said fourth hash against said decrypted third hash, said comparing means comprising means for providing an indication of integrity of said received reply EDI acknowledgement message and non-repudiation of origin of said reply EDI acknowledgement message; whereby non-repudiation of receipt of said EDI interchange communication is established by said sender computer.					

DOCKET NO. INOV-0005

Claim 18

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 14 wherein said EDI acknowledgement message comprises an AUTACK message.	AUTACK is a known EDI acknowledgement message. Col. 5, lines18-20.	UN 2, page 4; UN 3, page 5			

Claim 19

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 18 wherein said reply EDI acknowledgement message comprises an AUTACK message.	AUTACK is a known EDI acknowledgement message. Col. 5, lines18-20.	UN 2, page 4; UN 3, page 5			

DOCKET NO. INOV-0005

Claim 20

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 14 wherein said reply EDI acknowledgement message comprises an AUTACK message.	AUTACK is a known EDI acknowledgement message. Col. 5, lines18-20.	UN 2, page 4; UN 3, page 5			

Claim 21

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 1 wherein said EDI acknowledgement message comprises an AUTACK message.	AUTACK is a known EDI acknowledgement message. Col. 5, lines18-20.	UN 2, page 4; UN 3, page 5			

DOCKET NO. INOV-0005

Claim 22

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 21 wherein said open public network comprises the Internet.	EDI on internet is admitted to be known		Internet email is used for EDI, security needed. See p. 8	PGP used for internet email	

Claim 23

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 22 wherein each of said hashes comprise an MD5.	MD5 hash is conventional. Col. 5, lines 11-16.	MD5 is a known hash algorithm in EDI. UN 3, page 21 (MD5 listed as hash algorithm)		MD5 hash used in PGP.	

Claim 24

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 1 wherein each of said hashes	MD5 hash is conventional. Col. 5, lines 11-16.	MD5 is a known hash algorithm in EDI. UN 3, page 21 (MD5 listed as hash algorithm)		MD5 hash used in PGP	

comprise an MD5.					

Claim 25

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 24 wherein said open public network comprises the Internet.	EDI on internet is admitted to be known		Internet email is used for EDI, security needed. See p.8	PGP is used for internet email	

Claim 26

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 24 where said EDI acknowledgement message comprises an AUTACK message.	AUTACK is a known EDI acknowledgement message. Col. 5, lines18-20.	UN 2, page 4; UN 3, page 5			

DOCKET NO. INOV-0005

Claim 27

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 26 wherein said public and private keys comprise an RSA type cryptographic communication system.	Patentees admit that using RSA public/private key encryption for providing secure messages over the internet was known. Col. 1, lines 13-23; col. 4, lines 48-56.	UN 3, page 32—RSA security used in EDI AUTACK		PGP uses RSA type security	Fig. 2 illustrates RSA type of security.

Claim 28

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 27 wherein said open public network comprises the Internet.	EDI on internet is admitted to be known		Internet email is used for EDI, security needed. See, p.8	PGP is used for internet email.	

DOCKET NO. INOV-0005

Claim 29

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 1 wherein said public and private keys comprise an RSA type cryptographic communication system.	Patentees admit that using RSA public/private key encryption for providing secure messages over the internet was known. Col. 1, lines 13-23; col. 4, lines 48-56.	UN 3, page 32—RSA security used in EDI AUTACK		PGP uses RSA type security	Fig. 2 illustrates RSA type of security

Claim 30

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 29 wherein said open public network comprises the Internet.	EDI on internet is admitted to be known		Internet email is used for EDI, security needed. See p. 8.	PGP is used in internet email.	

DOCKET NO. INOV-0005

Claim 31

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 1 further comprising means for generating a trading partner agreement communication between said sender computer and said recipient computer, said sender computer and said recipient computer comprising trading partners, said trading partner agreement communication comprising said public keys in said EDI interchange communication for enabling said ttrading [sic] partners to provide certification to each other.		UN 1, page 31. .			

DOCKET NO. INOV-0005

Claim 32

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 31 wherein said open public network comprises the Internet.	EDI on internet is admitted to be known		Internet email is used for EDI, security needed. See p. 8.	PGP is used in Internet email.	

Claim 33

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with claim 32 wherein said EDI acknowledgement message comprises an AUTACK message.	AUTACK is a known EDI acknowledgement message. Col. 5, lines18-20.	UN 2, page 4, UN 3, page 5			

Claim 34

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
An improved secure open network communication system in accordance with	AUTACK is a known EDI acknowledgement message. Col. 5, lines18-20.	UN 2, page 4, UN 3, page 5			

claim 31 wherein said EDI acknowledgement message comprises an AUTACK message.					

Claim 35

	Admitted prior art	**UN docs**	**RFC**	**Byte**	**Sudia**
A method for selectively interconnecting a plurality of computers over an open public network for providing a computer exchange of private secure digital messages between a sender computer and a recipient computer in said plurality of computers,	Claim 1 is in Jepson format. As such the limitations in the preamble are admitted prior art. *In re Fout,* 675 F.2d 297, 301 (CCPA 1982); 37 CFR 1. 75(e). While claim 35 is not in Jepson format, the admissions made by patentees in regard to claim 1 apply to claim 35.	Each of UN 1, UN 2 and UN 3 describe EDI. UN 3, page 5.	Internet email useful for EDI, security needed. See p. 8.	BYTE describes Pretty Good Privacy (PGP) system for email security.	Sudia describes various schemes for encrypting electronic documents including using symmetric encryption (Fig. 1a), asymmetric encryption (Fig. 1g) and hashes with public key/private key (Fig. 2)
said sender computer having a first associated public key and a first associated private key,	RSA security known			PGP uses RSA (public key/private key	Fig. 2
said recipient computer having a second associated public key and a second associated private				PGP uses RSA (public key/private key	Fig. 2

key,					
said digital messages comprising an EDI interchange communication between said sender computer and said recipient computer, said EDI interchange communication having an associated EDI acknowledgment message	Patentees admit AUTACK known EDI acknowledgement message. Col. 5, lines 18-20.	UN 1, page 8 describes the use of security schemes applied to the message including integrity and non-repudiation of origin (NRO). UN 2, page 6 in discussing Segment Group 1: USH-USA-SG2 states "A group of segments providing all security information necessary for the integrity, authentication, and non-repudiation of origin or of receipt of all entities referenced in the AUTACK message and the AUTACK message itself. EDI messages including AUTACK may be secured using hashes, e.g., MD5, and public key/private key security system, e.g., RSA. UN 1, page 15, code 6 MD5 and RSA.			

		AUTACK can be used by sender as authentication message and by the recipient as an acknowledgement message.			
said method comprising the steps of digitally signing said associated EDI acknowledgement message with said sender computer's private key;	Patentees admit that using RSA public key/private key encryption for providing secure messages over the internet was known. Col. 1, lines 13-23; col. 4, lines 48-56.	RSA public key/private key security used in AUTACK, UN 3, page 32.		PGP uses RSA public key/private key security	Fig. 2
transmitting said EDI interchange communication along with said digitally signed associated EDI acknowledgement message to said recipient computer over said open public network;	Patentees admit that EDI was transacted on the internet			MIME email used to send EDI over internet	
and processing said received digitally signed EDI acknowledgement message for providing authentication and non-repudiation of said EDI interchange communication from said sender		E.g., UN 2, page 4, UN 3, page 5, AUTACK used to authenticate integrity of content and non-repudiation of origin when used by the sender.			

computer,					
said processing step comprising the step of processing said received digitally signed associated EDI acknowledgement message with said sender's public key;		RSA public key/private key security used in AUTACK, UN 3, page 32.		PGP uses public key/private key security	Fig. 2
whereby secure private EDI interchange communications can occur over an open public network while providing authentication and non-repudiation of said EDI communications using said associated EDI acknowledgement message.		E.g., UN 2, page 4, UN 3, page 5 AUTACK used to authenticate integrity of content and non-repudiation of origin when used by the sender.			

DOCKET NO. INOV-0005

Claim 36

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 35 wherein said open public network comprises the Internet.	EDI on internet is admitted to be known		Internet email is used for EDI, security needed. See p. 8.	PGP is used on internet email	

Claim 37

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 36 further comprising the steps of creating a reply EDI acknowledgement message from said recipient computer; digitally signing said reply EDI acknowledgement message with said recipient computer's private key; transmitting said digitally signed	AUTACK admitted known Use of public key/private key security is admittedly known.	RSA public key/private key security used on EDI messages. UN 1, Examples 2 and 3, pages 39-48.		PGP is used on internet email	See Fig. 2.

reply EDI acknowledgement message to said sender computer over said open public network, said sender computer receiving said digitally signed reply EDI acknowledgement message; and processing said received digitally signed reply EDI acknowledgement message for providing non-repudiation of receipt of said EDI interchange communication by said sender computer, said processing step comprising the step of processing said received digitally signed reply EDI acknowledgement message with said recipient computer's public key; whereby non-repudiation of receipt of said EDI interchange communication is established by said sender computer.		UN 2, page 4 and UN 3, page 5 state that a secure AUTACK sent from the recipient to the sender is non-repudiation of receipt.			

DOCKET NO. INOV-0005

Claim 38

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 35 further comprising the steps of creating a reply EDI acknowledgement message from said recipient computer; digitally signing said reply EDI acknowledgement message with said recipient computer's private key; transmitting said digitally signed reply EDI acknowledgement message to said sender computer over said open public network, said sender computer receiving said digitally signed reply EDI acknowledgement message; and processing said received digitally signed reply EDI acknowledgement message for	AUTACK admittedly known Use of public key/private key security is admittedly known.	RSA public key/private key security used on EDI messages. UN 1, Examples 2 and 3, pages 39-48. UN 2, page		PGP is used on internet email	See Fig. 2

providing non-repudiation of receipt of said EDI interchange communication by said sender computer, said processing step comprising the step of processing said received digitally signed reply EDI acknowledgement message with said recipient computer's public key; whereby non-repudiation of receipt of said EDI interchange communication is established by said sender computer.		4 and UN 3, page 5 state that a secure AUTACK sent from the recipient to the sender is non-repudiation of receipt.			

Claim 39

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 38 wherein said processing step further comprises the step of providing non-repudiation of origin at said recipient computer from said received		UN 2, page 4 and UN 3, page 5 state that a secure AUTACK sent from the sender to the recipient is non-repudiation of origin.			

EDI a[c]knowledgement message.					

Claim 40

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 39 wherein said open public network comprises the Internet.	EDI on internet is admitted to be known		Internet email used with EDI, security needed. See p. 8.	PGP security used on internet email.	

Claim 41

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 41 wherein said EDI acknowledgement message comprises an AUTACK message.	Patentees admit that AUTACK is a known EDI acknowledgement message. Col. 5, lines18-20.	UN 2, page 4; UN 3, page 5			

DOCKET NO. INOV-0005

Claim 43

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 35 wherein said processing step further comprises the step of providing non-repudiation of origin at said recipient computer from said received EDI a[c]knowledgement message.		UN 2, page 4 and UN 3, page 5 state that a secure AUTACK sent from the sender to the recipient is non-repudiation of origin.			

Claim 44

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 35 wherein said EDI acknowledgement message comprises an AUTACK message.	Patentees admit that AUTACK is a known EDI acknowledgement message. Col. 5, lines18-20.	UN 2, page 4; UN 3, page 5			

Claim 45

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 35 wherein said public and private keys comprise an RSA type cryptographic communication system.	Patentees admit that using RSA public/private key encryption for providing secure messages over the internet was known. Col. 1, lines 13-23; col. 4, lines 48-56.	UN 3, page 32—RSA security used in EDI AUTACK		PGP uses RSA public key/private key security.	See Fig. 2.

Claim 46

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 45 wherein said open public network comprises the Internet.	EDI on internet is admitted to be known		Internet email is used for EDI, security needed. See p. 8	PGP security used on internet email	

DOCKET NO. INOV-0005

Claim 47

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 45 wherein said EDI acknowledgement message comprises an AUTACK message	Patentees admit that AUTACK is a known EDI acknowledgement message. Col. 5, lines18-20.	UN 2, page 4; UN 3, page 5			

Claim 48

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 47 further comprising the steps of creating a reply EDI acknowledgement message from said recipient computer; digitally signing said reply EDI acknowledgement message with said recipient computer's private key; transmitting said	AUTACK admittedly known Use of public key/private key security is admittedly known.	RSA public key/private key security used on EDI messages. UN 1, Examples 2 and 3, pages 39-48.		PGP is used on internet email	See Fig. 2.

digitally signed reply EDI acknowledgement message to said sender computer over said open public network, said sender computer receiving said digitally signed reply EDI acknowledgement message; and processing said received digitally signed reply EDI acknowledgement message for providing non-repudiation of receipt of said EDI interchange communication by said sender computer, said processing step comprising the step of processing said received digitally signed reply EDI acknowledgement message with said recipient computer's public key; whereby non-repudiation of receipt of said EDI interchange communication is established by said sender computer.		UN 2, page 4 and UN 3, page 5 state that a secure AUTACK sent from the recipient to the sender is non-repudiation of receipt.			

DOCKET NO. INOV-0005

Claim 49

	Admitted prior art	**Un docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 48 wherein said reply EDI acknowledgement message comprises an an AUTACK message.	Patentees admit that AUTACK is a known EDI acknowledgement message. Col. 5, lines18-20.	UN 2, page 4; UN 3, page 5			

Claim 50

	Admitted prior art	**UN docs**	**RFC**	**BYTE**	**Sudia**
A method for providing secure private communications over an open public network in accordance with claim 49 wherein said open public network comprises the Internet.	EDI on internet is admitted to be known		Internet email is used for EDI, security needed. See p. 8.	PGP security used on internet email.	

DOCKET NO. INOV-0005

CONCLUSION

The claims of the '669 patent are not novel or non-obvious when considered in view of the references as set forth herein. Accordingly, Requestor respectfully requests that this Request be granted.

Dated: April 14, 2008

Woodcock Washburn LLP

By /Lance D. Reich/
Lance D. Reich
Registration No. 42,097

2929 Arch Street
Cira Centre, 12th Floor
Philadelphia, PA 19104-2891
(215) 564-5739
(215) 568-3439

DOCKET NO. INOV-0005

Certificate of Service in Compliance With 37 C.F.R. § 1.510(b)(5)

The undersigned certifies that copies of the following:

(1) Request for *Ex Parte* Reexamination of Patent Transmittal Form; and

(2) Request for *Ex Parte* Reexamination including Exhibits,

were served on the purported owner of the Patent at the correspondence address shown in PAIR (as required under 37 C.F.R. § 1.33(c)):

PERKINS & MITTNER, LLP
750 B STREET, SUITE 2800
SAN DIEGO CA 92101

in accordance with 37 C.F.R. § 1.510(b)(5) on the 14th day of April, 2008.

Dated: April 14, 2008

Woodcock Washburn LLP

By /Lance D. Reich/
Lance D. Reich
Registration No. 42,097

2929 Arch Street
Cira Centre, 12th Floor
Philadelphia, PA 19104-2891
(215) 564-5739
Fax: (215) 568-3439

PTO/SB/57 (09-07)
Approved for use through 08/31/2010. OMB 0651-0033
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

(Also referred to as FORM PTO-1465)

REQUEST FOR *EX PARTE* REEXAMINATION TRANSMITTAL FORM

Address to:
Mail Stop *Ex Parte* Reexam
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Attorney Docket No.: INOV-0005

Date: April 14, 2008

1. [X] This is a request for *ex parte* reexamination pursuant to 37 CFR 1.510 of patent number 5,812,669 issued 9/22/1998. The request is made by:

   [ ] patent owner. [X] third party requester.

2. [X] The name and address of the person requesting reexamination is:

   Inovis USA, Inc.

   11720 AmberPark Drive

   Alpharetta, GA 30004

3. [ ] a. A check in the amount of $_____ is enclosed to cover the reexamination fee, 37 CFR 1.20(c)(1);

   [X] b. The Director is hereby authorized to charge the fee as set forth in 37 CFR 1.20(c)(1) to Deposit Account No. 23-3050 (submit duplicative copy for fee processing); or

   [ ] c. Payment by credit card. Form PTO-2038 is attached.

4. [X] Any refund should be made by [ ] check or [X] credit to Deposit Account No. 23-3050. 37 CFR 1.26(c). If payment is made by credit card, refund must be to credit card account.

5. [X] A copy of the patent to be reexamined having a double column format on one side of a separate paper is enclosed. 37 CFR 1.510(b)(4)

6. [ ] CD-ROM or CD-R in duplicate, Computer Program (Appendix) or large table
   [ ] Landscape Table on CD

7. [ ] Nucleotide and/or Amino Acid Sequence Submission
   If applicable, items a. – c. are required.

   a. [ ] Computer Readable Form (CRF)
   b. Specification Sequence Listing on:

      i. [ ] CD-ROM (2 copies) or CD-R (2 copies); **or**
      ii. [ ] paper

   c. [ ] Statements verifying identity of above copies

8. [ ] A copy of any disclaimer, certificate of correction or reexamination certificate issued in the patent is included.

9. [X] Reexamination of claim(s) 1-50 is requested.

10. [X] A copy of every patent or printed publication relied upon is submitted herewith including a listing thereof on Form PTO/SB/08, PTO-1449, or equivalent.

11. [ ] An English language translation of all necessary and pertinent non-English language patents and/or printed publications is included.

PTO/SB/57 (09-07)
Approved for use through 08/31/2010. OMB 0651-0033
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

12. [X] The attached detailed request includes at least the following items:

a. A statement identifying each substantial new question of patentability based on prior patents and printed publications. 37 CFR 1.510(b)(1)
b. An identification of every claim for which reexamination is requested, and a detailed explanation of the pertinency and manner of applying the cited art to every claim for which reexamination is requested. 37 CFR 1.510(b)(2)

13. [ ] A proposed amendment is included (only where the patent owner is the requester). 37 CFR 1.510(e)

14. [X] a. It is certified that a copy of this request (if filed by other than the patent owner) has been served in its entirety on the patent owner as provided in 37 CFR 1.33(c).
The name and address of the party served and the date of service are:

Perkins & Mittner, LLP

750 B Street, Suite 2800

San Diego, CA 92101

Date of Service: Apri 14, 2008; or

[ ] b. A duplicate copy is enclosed since service on patent owner was not possible.

15. Correspondence Address: Direct all communication about the reexamination to:

[ ] The address associated with Customer Number: 23377

OR

[ ] Firm or Individual Name

Address

City	State	Zip
Country		
Telephone	Email	

16. [ ] The patent is currently the subject of the following concurrent proceeding(s):
[ ] a. Copending reissue Application No. _____.
[ ] b. Copending reexamination Control No. _____.
[ ] c. Copending Interference No. _____.
[ ] d. Copending litigation styled:

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

/Lance D. Reich/ — Authorized Signature
April 14, 2008 — Date

Lance D. Reich — Typed/Printed Name
42,097 — Registration No.
[ ] For Patent Owner Requester
[X] For Third Party Requester

EXHIBIT B-1

US005659616A

United States Patent [19]

Sudia

[11] Patent Number: 5,659,616

[45] Date of Patent: Aug. 19, 1997

[54] **METHOD FOR SECURELY USING DIGITAL SIGNATURES IN A COMMERCIAL CRYPTOGRAPHIC SYSTEM**

[75] Inventor: **Frank Wells Sudia**, New York, N.Y.

[73] Assignee: **Certco, LLC**, New York, N.Y.

[21] Appl. No.: **682,071**

[22] Filed: **Jul. 16, 1996**

Related U.S. Application Data

[63] Continuation of Ser. No. 277,438, Jul. 19, 1994, abandoned.

[51] **Int. Cl.**[6] **H04K 1/00**

[52] **U.S. Cl.** **380/23**; 380/25; 380/30

[58] **Field of Search** 380/23, 25, 24, 380/4, 3, 49, 29, 30

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,625,076	11/1986	Okamoto et al.	380/23
4,981,370	1/1991	Dziewit et al.	380/25
5,005,200	4/1991	Fischer	380/30
5,031,214	7/1991	Dziewit et al.	380/23
5,157,726	10/1992	Merkle et al.	380/23
5,163,091	11/1992	Graziano et al.	380/25
5,191,613	3/1993	Graziano et al.	380/215
5,214,702	5/1993	Fischer	380/30

OTHER PUBLICATIONS

ANSI X9.30–199x (Working Draft) Public Key Cryptography Using Irreversible Algorithms for the Financial Services Industry: Part 3: Certificate Management for DSA, Jun. 1, 1994, pp. i–86.

Secure Data Network System, Access Control Specification, Access Control Information Specification (ACIS) Addendum 1 (SDN.802/1), Jul. 25, 198 pp. ii–85.

Secure Data Network System, Access Control Specification, SDN.802, Rev. 1.0 Jul. 25, 1989, pp. 1.43.

Secure Data Network System; Access Control Concept Document (Revision 1.3), SDN.801, Jul. 26, 1989, pp. 1–18.

European Computer Manufacturers Association, Standard ECMA–138 Security in Open Systems —Data Elements and Service Definitions, Dec. 1989, pp. i–81.

Addison Fischer, Workflow. 2000–Electronic Document Authorization in Practice, Fischer International Systems Corporation, Copyright 1992, 7 pages.

Richard Ankney, Certificate Management for the Financial Services Industry, Aba/Scitech/Notaization and Nonrepudiation WG. Mtg. of Jul. 1, 1993.

ANSI X9.30 (Working Draft) Public Key Cryptography Using Irreversible Algorithms for the Financial Services Industry: Part 3: Certificate Management for DSA, Mar. 29, 1993, pp. i–71.

ANSI X9.30–199x (Working Draft) Public Key Cryptography Using Irreversible Algorithms for the Financial Services Industry: Part 3: Certificate Management of DSA, Sep. 27, 1993, p. i–87.

Rich Ankney, et al. Enhanced Management Controls Using Attribute Certificates, ASC X9 Project Proposal No. X9F–1–3, Nov. 10, 1993, 13 pages.

(List continued on next page.)

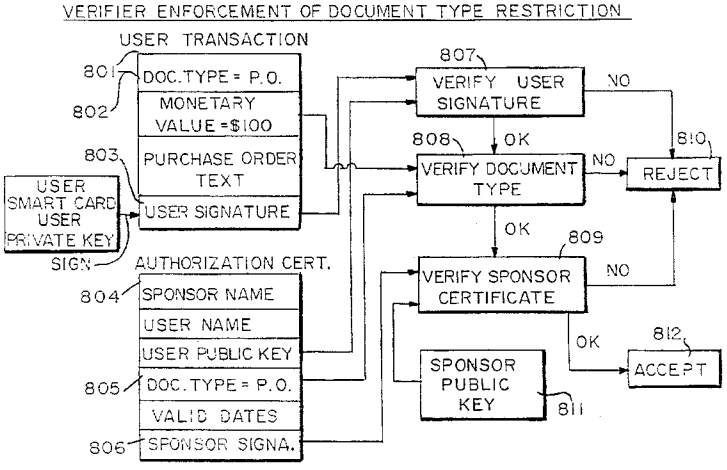
Primary Examiner—David C. Cain

Attorney, Agent, or Firm—Cushman, Darby & Cushman IP Group of Pillsbury Madison & Sutro LLP

[57] **ABSTRACT**

A system for securely using digital signatures in a commercial cryptographic system that allows industry-wide security policy and authorization information to be encoded into the signatures and certificates by employing attribute certificates to enforce policy and authorization requirements. In addition to value limits, cosignature requirements and document type restrictions that can be placed on transactions, an organization can enforce with respect to any transaction geographical and temporal controls, age-of-signature limitations, preapproved counterparty limitations and confirm-to requirements by using attribute certificates for the transacting user. Restrictions on distribution of certificates can be set using attribute certificates. Certificates can be used also to ensure key confinement and non-decryption requirements of smartcards in this system.

37 Claims, 15 Drawing Sheets



OTHER PUBLICATIONS

ANSI X9.30–199x (Draft), Executive Summary, Public Key Cryptography Using Irreversibl Algorithms for the Financial Services Industry: Part 3: Certificate Management for DSA Nov. 18, 1993, pp. 1–6.

ANSI X9.xx–19x (Working Draft) Enhanced Management Controls Using Attribute Certificates, Jan. 3, 1994, pp. 1–18.

ANSI X9.30–199X (Working Draft) Public Key Cryptography Using Reversible Algorithms for the Financial Services Industry: Part 3.

PKCS #7: Cryptographic Message Syntax Standard, Version 1.4, Jun. 3, 1991, pp. 1–24.

Recommendation X 500: The Directory–Overview of Concepts, Models and Services, Melbourne, 1988, pp. 1–13.

Robert Jueneman, Limiting The Liability of CAs and Individuals Regarding the Use of Digital Signatures, Jun. 30, 1993, pp. 1–8.

John Linn, Practical Authentication for Distributed Computing, IEEE, 1990, pp. 32–40.

Morrie Gasser et al, An Architecture for Practical Delegation in a Distributed System, IEEE, 1990, pp. 20–30.

Denis Pinkas et al. Sesame: Secure European System for Applications in a Multivendor Environment, Issue 1, Feb. 1993.

J. Linn, Privacy Enhancement for Internet Electronic Mail: Part I, Feb. 1993, pp. 1–42.

S. Kent, Privacy Enhancement for Internet Electronic Mail: Part II, Feb. 1993, pp. 1–32.

X1258 Version 2, ASC X–12 Draft Standard . . . Managing Electronic Data Interchange, pp. 1–40.

Financial Institution Sign–On Authentication for Wholesale Financial Transactions X926, Approved:Feb. 28,1990, pp. 1–25.

Draft ANSI X9 30–199x, Public Key Cryptography Using Irreversible Algorithms for the Financial Services Industry: Part 3, Nov. 18, 1993, pp. 1–6.

PKCS #7: Cryptographic Message Syntax Standard Version 15, Revised Nov. 1, 1993.

Information Technology —Open Systems Interconnection–The Directory: Authentication Framework —Recommendation X 509 ISO/IEC 9594–8 (1993), pp. i–35.

Accredited Standards Committee X9, X9–Financial Services, Public Key Cryptography Using Irreversible Algorithms for the Financial Services Industry: Part 3, Oct. 7, 1995, pp. i–81.

Frank Sudia and Richard Ankney, Commercialization of Digital Signatures, July 20 Boston, pp. 1–16.

Addison M. Fischer, Electronic Document Authorization, National Computer Security Conference, 1992, pp. 1–23.

ECMA —Standard ECMA–138 —Security In Open Systems —Data Elements and Service Definitions, Dec. 1989, pp. i–81.

ECMA —Security in Open Systems A Security Framework, ECMA TR/46, Jul. 1988, pp. i–71.

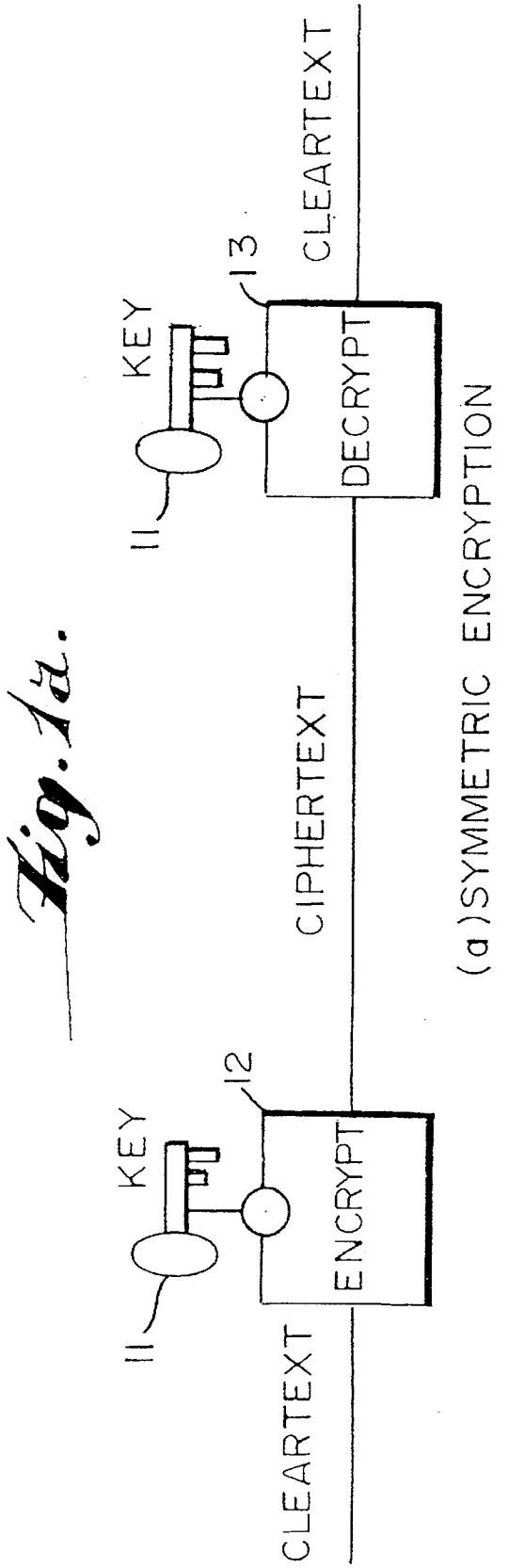

KEY
11
13
DECRYPT
CLEARTEXT
CIPHERTEXT
KEY
11
12
ENCRYPT
CLEARTEXT
(a) SYMMETRIC ENCRYPTION

Fig. 1a.

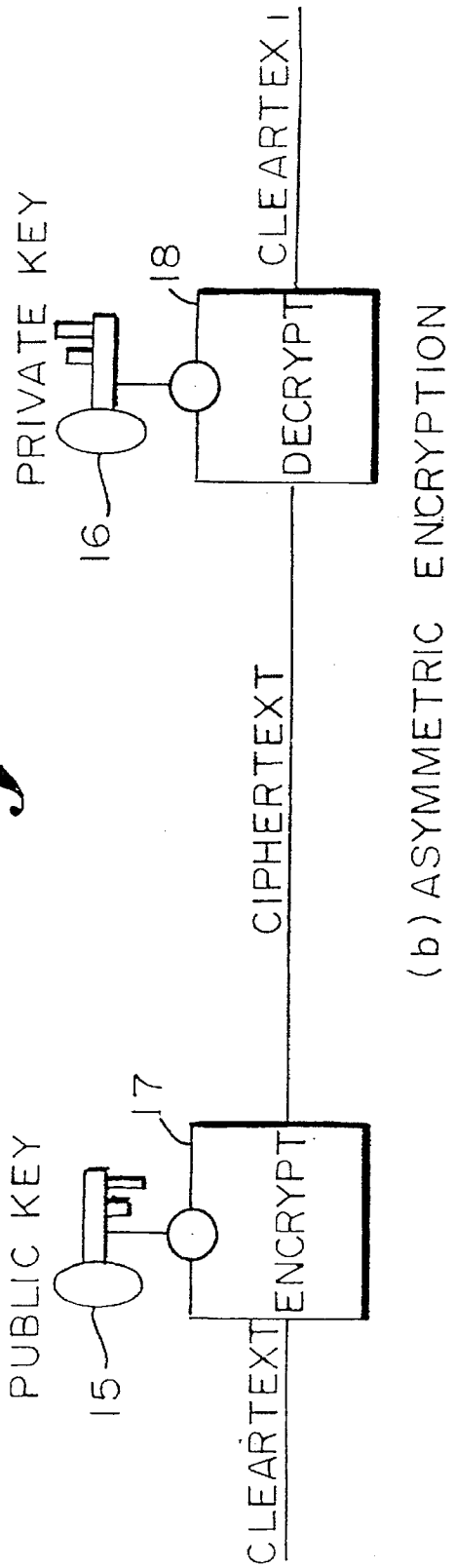
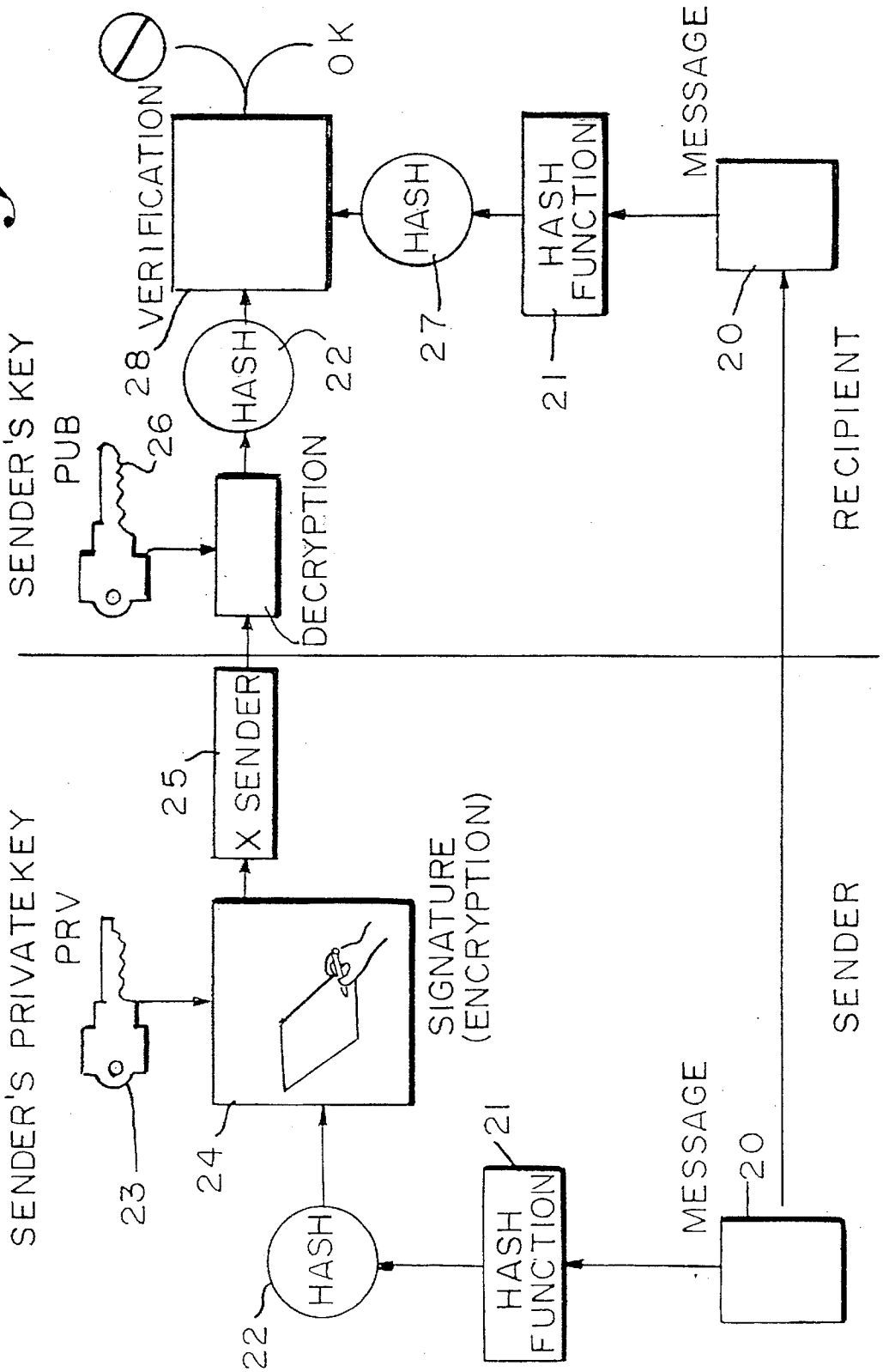

PRIVATE KEY
16
18
DECRYPT
CLEARTEXT
CIPHERTEXT
PUBLIC KEY
15
17
ENCRYPT
CLEARTEXT
(b) ASYMMETRIC ENCRYPTION

Fig. 1b.

Fig. 2.


SENDER'S KEY
PUB
26
DECRYPTION
HASH
22
28
VERIFICATION
OK
HASH
27
HASH FUNCTION
21
MESSAGE
20
RECIPIENT
SENDER'S PRIVATE KEY
PRV
23
25
X SENDER
24
SIGNATURE (ENCRYPTION)
22
HASH
21
HASH FUNCTION
MESSAGE
20
SENDER

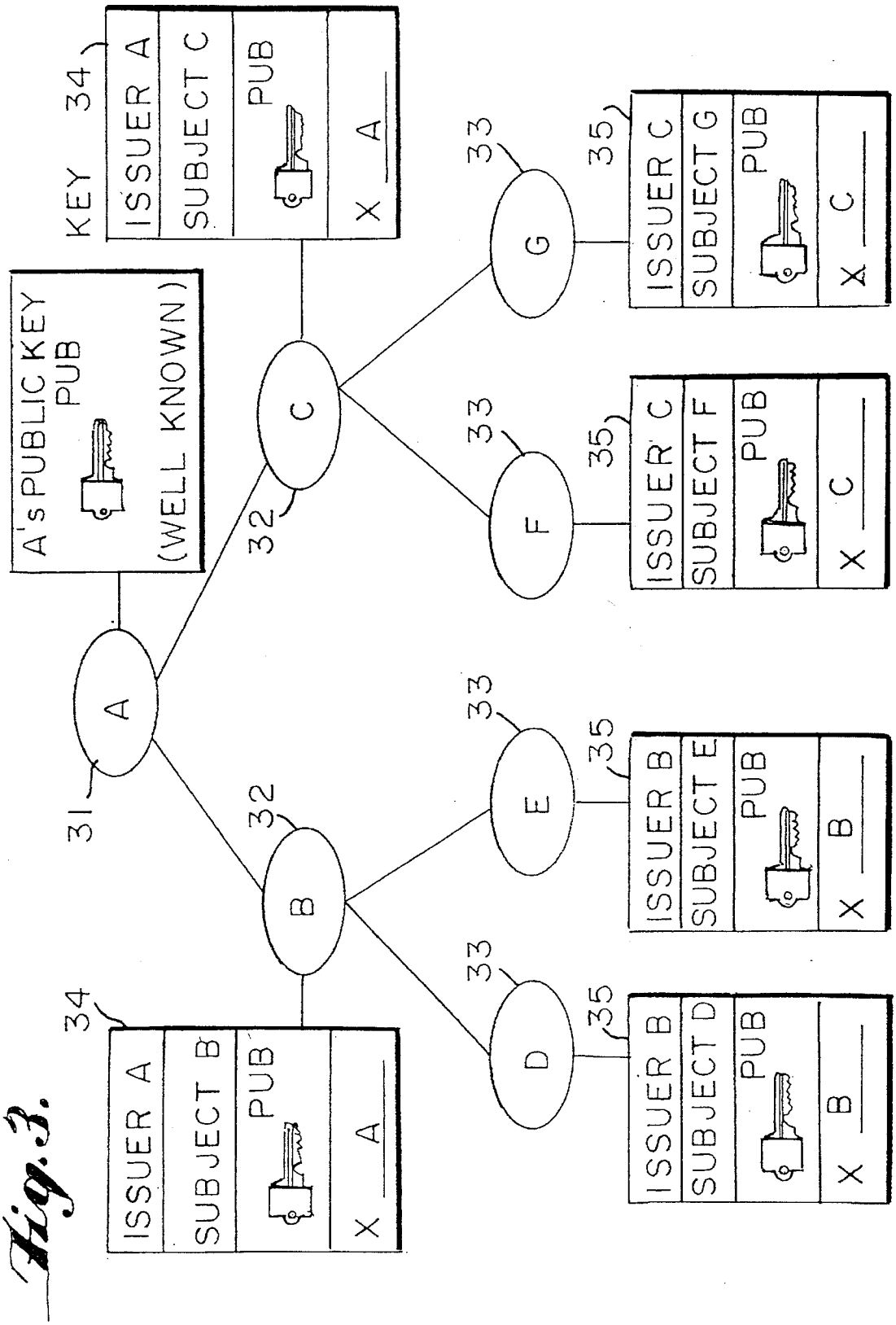

KEY 34
ISSUER A
SUBJECT C
PUB
X A
A's PUBLIC KEY PUB (WELL KNOWN)
31
A
32
C
32
B
33
G
33
F
33
E
33
D
35
ISSUER C
SUBJECT G
PUB
X C
35
ISSUER C
SUBJECT F
PUB
X C
35
ISSUER B
SUBJECT E
PUB
X B
35
ISSUER B
SUBJECT D
PUB
X B
34
ISSUER A
SUBJECT B
PUB
X A

Fig. 3.

Fig. 4.

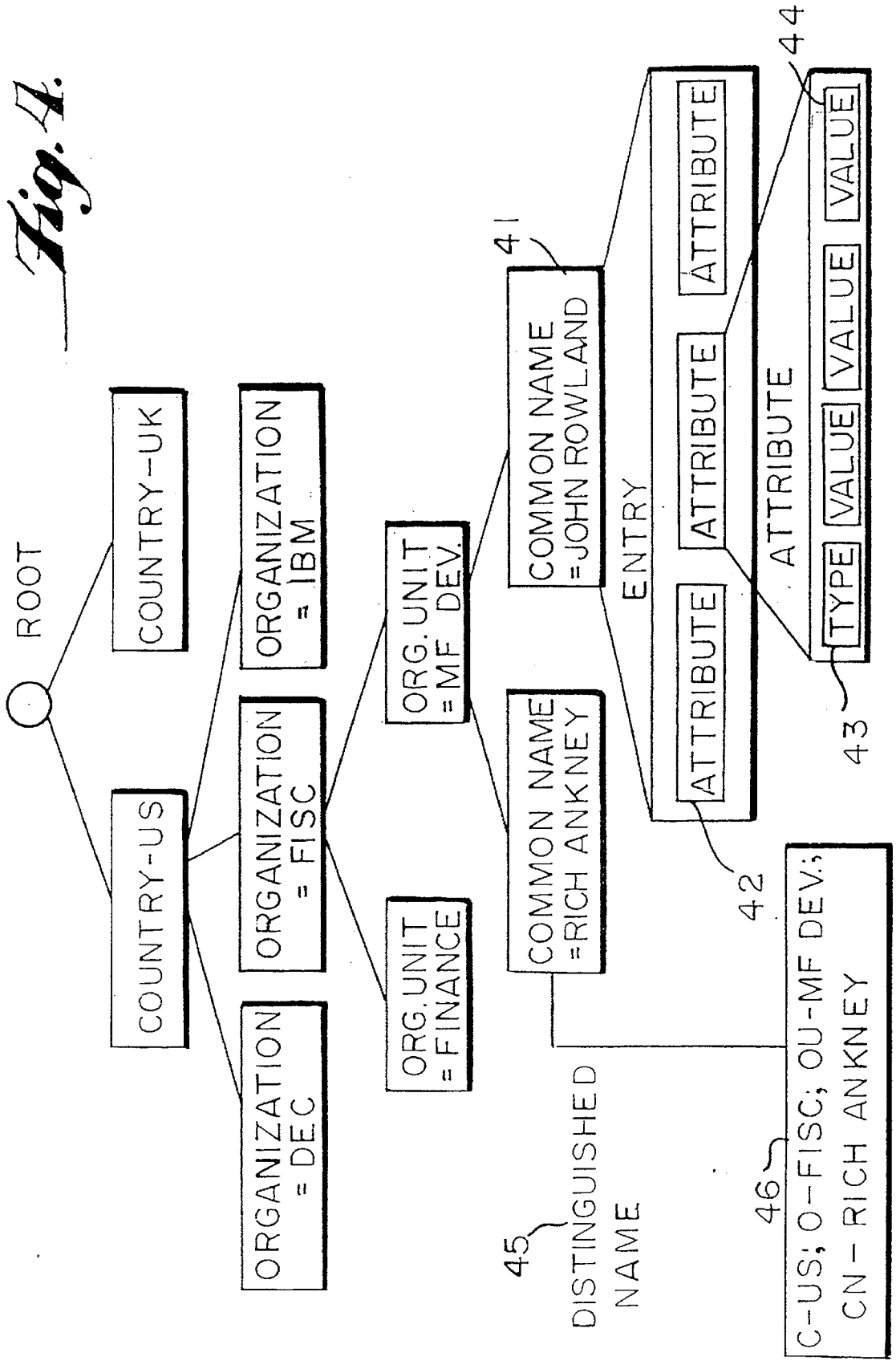
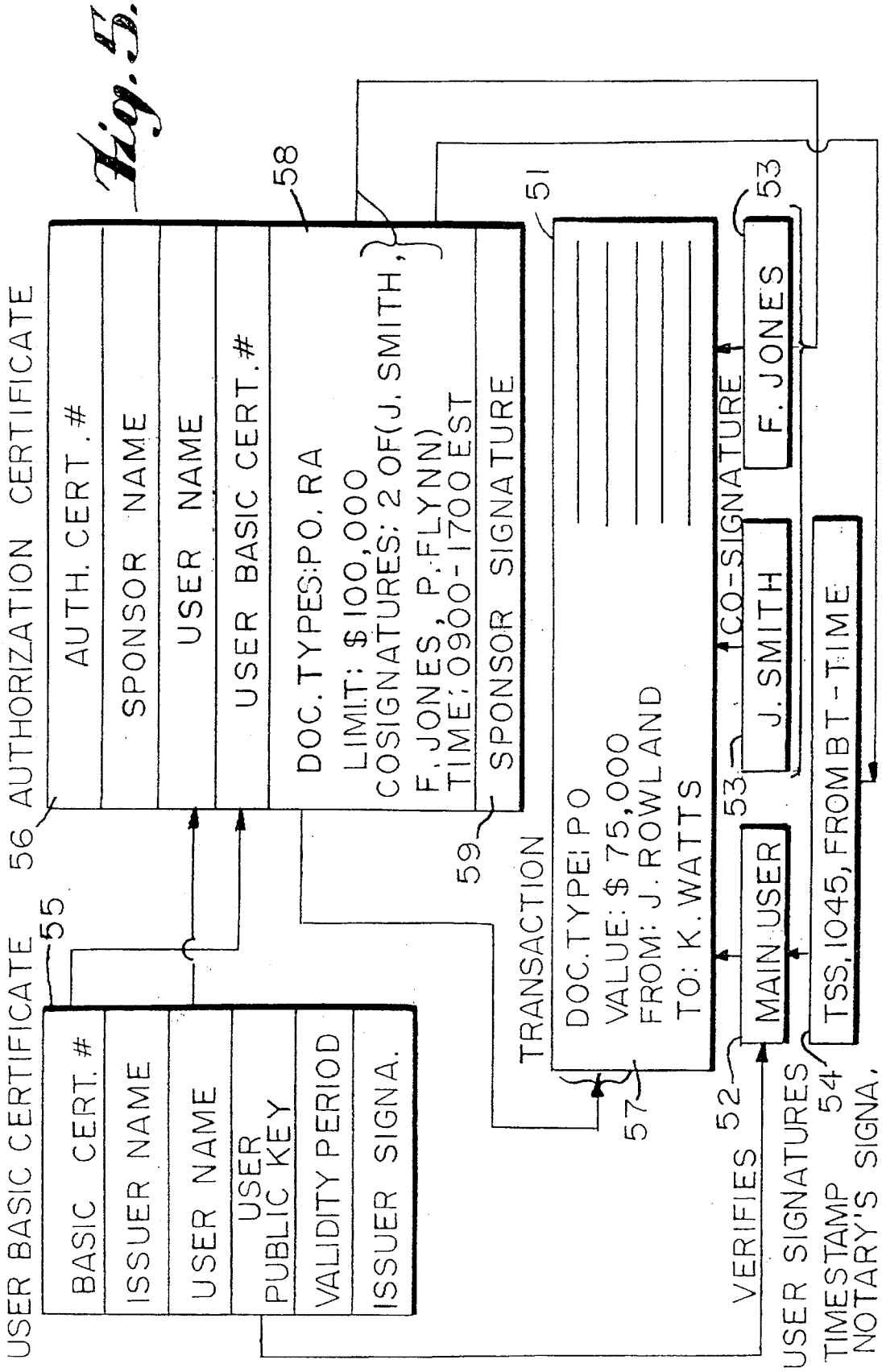

ROOT
COUNTRY-UK
COUNTRY-US
ORGANIZATION = IBM
ORGANIZATION = FISC
ORGANIZATION = DEC
ORG.UNIT = MF DEV.
ORG.UNIT = FINANCE
COMMON NAME = JOHN ROWLAND
COMMON NAME = RICH ANKNEY
41
ENTRY
ATTRIBUTE
ATTRIBUTE
ATTRIBUTE
42
ATTRIBUTE
TYPE
VALUE
VALUE
VALUE
44
43
45
DISTINGUISHED NAME
46
C-US; O-FISC; OU-MF DEV.; CN - RICH ANKNEY

Fig. 5.


USER BASIC CERTIFICATE
BASIC CERT. #
ISSUER NAME
USER NAME
USER PUBLIC KEY
VALIDITY PERIOD
ISSUER SIGNA.
55
56 AUTHORIZATION CERTIFICATE
AUTH. CERT. #
SPONSOR NAME
USER NAME
USER BASIC CERT. #
DOC. TYPES: PO, RA
LIMIT: $100,000
COSIGNATURES: 2 OF (J. SMITH, F. JONES, P. FLYNN)
TIME: 0900-1700 EST
SPONSOR SIGNATURE
58
59
TRANSACTION
DOC. TYPE: PO
VALUE: $ 75,000
FROM: J. ROWLAND
TO: K. WATTS
51
57
VERIFIES
MAIN USER
52
CO-SIGNATURE
J. SMITH
F. JONES
53
USER SIGNATURES
TSS, 1045, FROM BT - TIME
54
TIMESTAMP
NOTARY'S SIGNA,

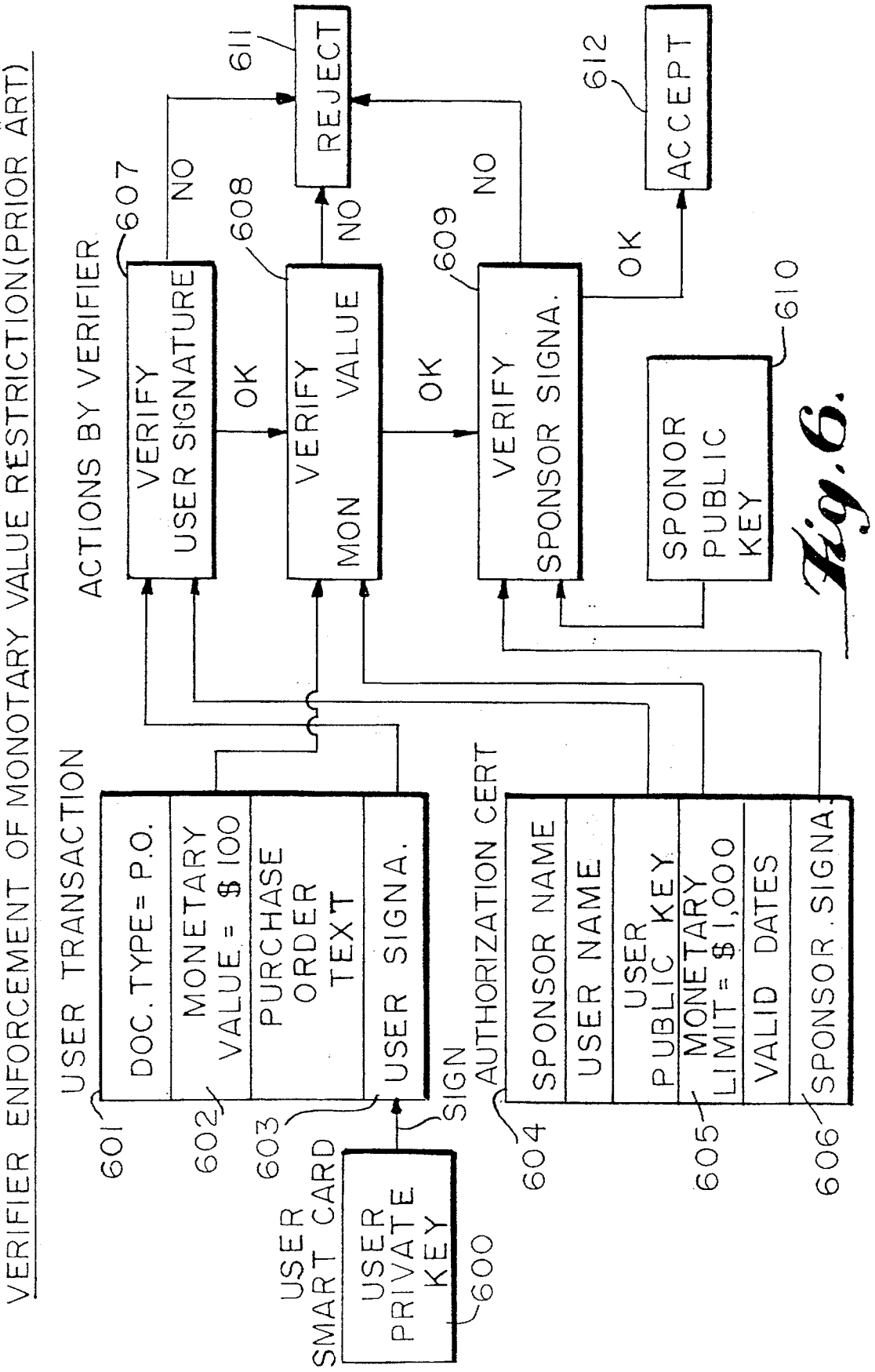

VERIFIER ENFORCEMENT OF MONOTARY VALUE RESTRICTION(PRIOR ART)
USER TRANSACTION
DOC. TYPE= P.O.
MONETARY VALUE = $100
PURCHASE ORDER TEXT
USER SIGNA.
601
602
603
SIGN
USER SMART CARD
USER PRIVATE KEY
600
AUTHORIZATION CERT
SPONSOR NAME
USER NAME
USER PUBLIC KEY
MONETARY LIMIT = $1,000
VALID DATES
SPONSOR. SIGNA.
604
605
606
ACTIONS BY VERIFIER
607
VERIFY USER SIGNATURE
608
VERIFY MON VALUE
609
VERIFY SPONSOR SIGNA.
610
SPONOR PUBLIC KEY
611
REJECT
612
ACCEPT
OK
NO
OK
NO
OK
NO

Fig. 6.

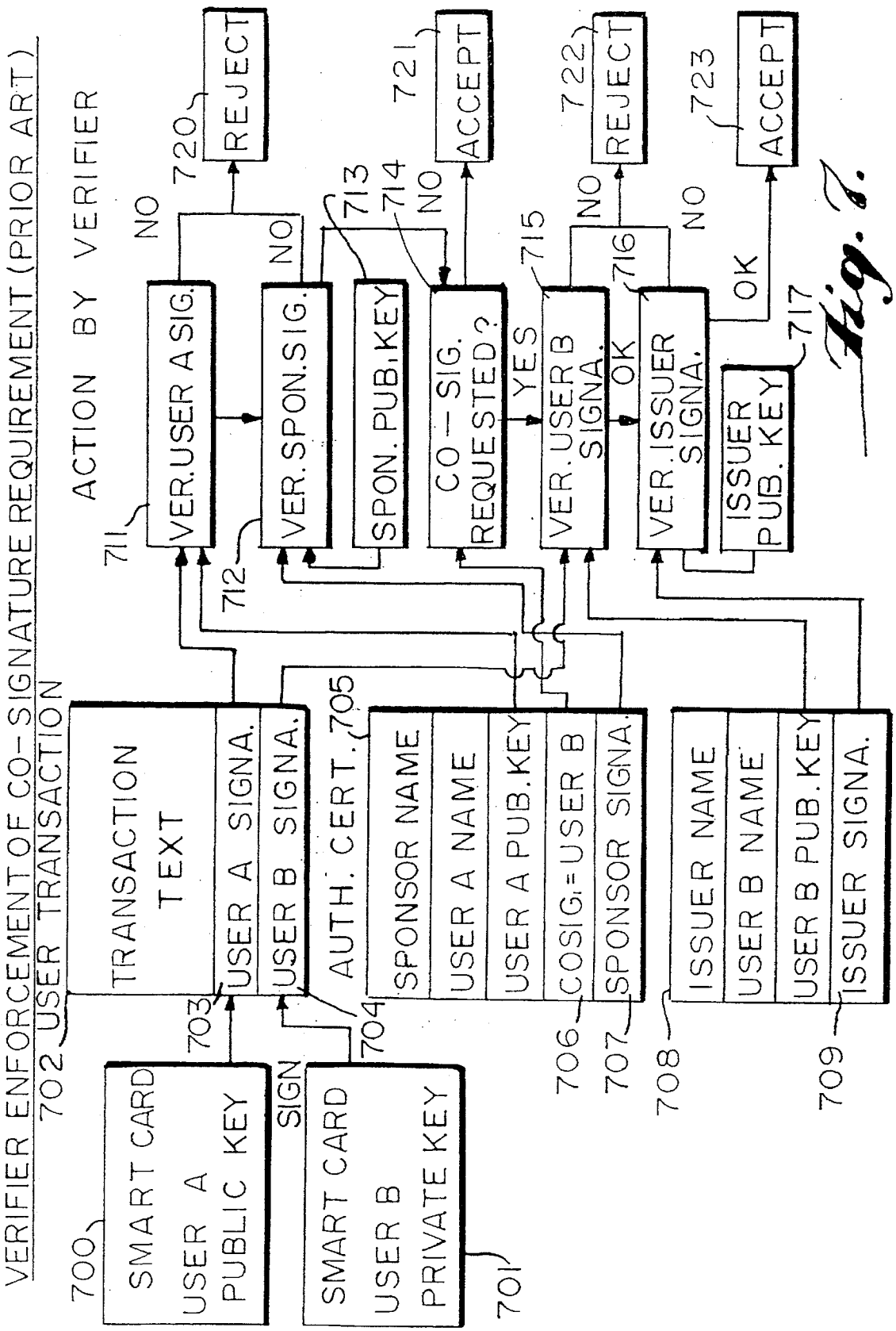

VERIFIER ENFORCEMENT OF CO-SIGNATURE REQUIREMENT (PRIOR ART)
ACTION BY VERIFIER
USER TRANSACTION
TRANSACTION TEXT
USER A SIGNA.
USER B SIGNA.
AUTH. CERT.
SPONSOR NAME
USER A NAME
USER A PUB.KEY
COSIG.=USER B
SPONSOR SIGNA.
ISSUER NAME
USER B NAME
USER B PUB.KEY
ISSUER SIGNA.
SMART CARD USER A PUBLIC KEY
SMART CARD USER B PRIVATE KEY
SIGN
VER.USER A SIG.
VER.SPON.SIG.
SPON.PUB.KEY
CO-SIG. REQUESTED?
VER.USER B SIGNA.
VER.ISSUER SIGNA.
ISSUER PUB. KEY
REJECT
ACCEPT
REJECT
ACCEPT
NO
NO
NO
NO
YES
OK
OK
700
701
702
703
704
705
706
707
708
709
711
712
713
714
715
716
717
720
721
722
723

Fig. 6.

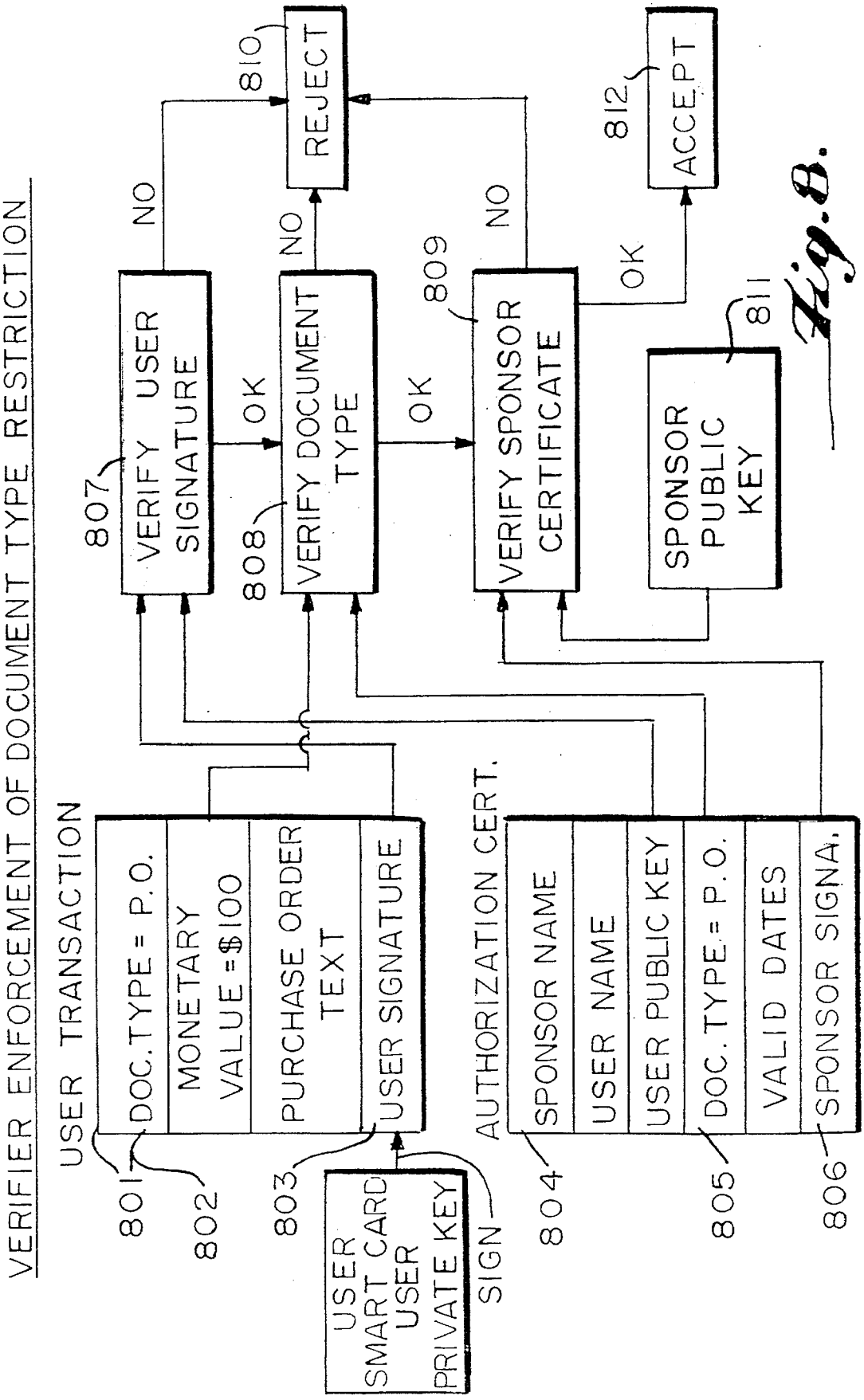

VERIFIER ENFORCEMENT OF DOCUMENT TYPE RESTRICTION
USER TRANSACTION
DOC. TYPE = P.O.
MONETARY VALUE = $100
PURCHASE ORDER TEXT
USER SIGNATURE
801
802
803
USER SMART CARD USER PRIVATE KEY
SIGN
AUTHORIZATION CERT.
SPONSOR NAME
USER NAME
USER PUBLIC KEY
DOC. TYPE = P.O.
VALID DATES
SPONSOR SIGNA.
804
805
806
807
VERIFY USER SIGNATURE
808
VERIFY DOCUMENT TYPE
809
VERIFY SPONSOR CERTIFICATE
SPONSOR PUBLIC KEY
811
810
REJECT
812
ACCEPT
NO
OK
NO
OK
NO
OK

Fig. 8.

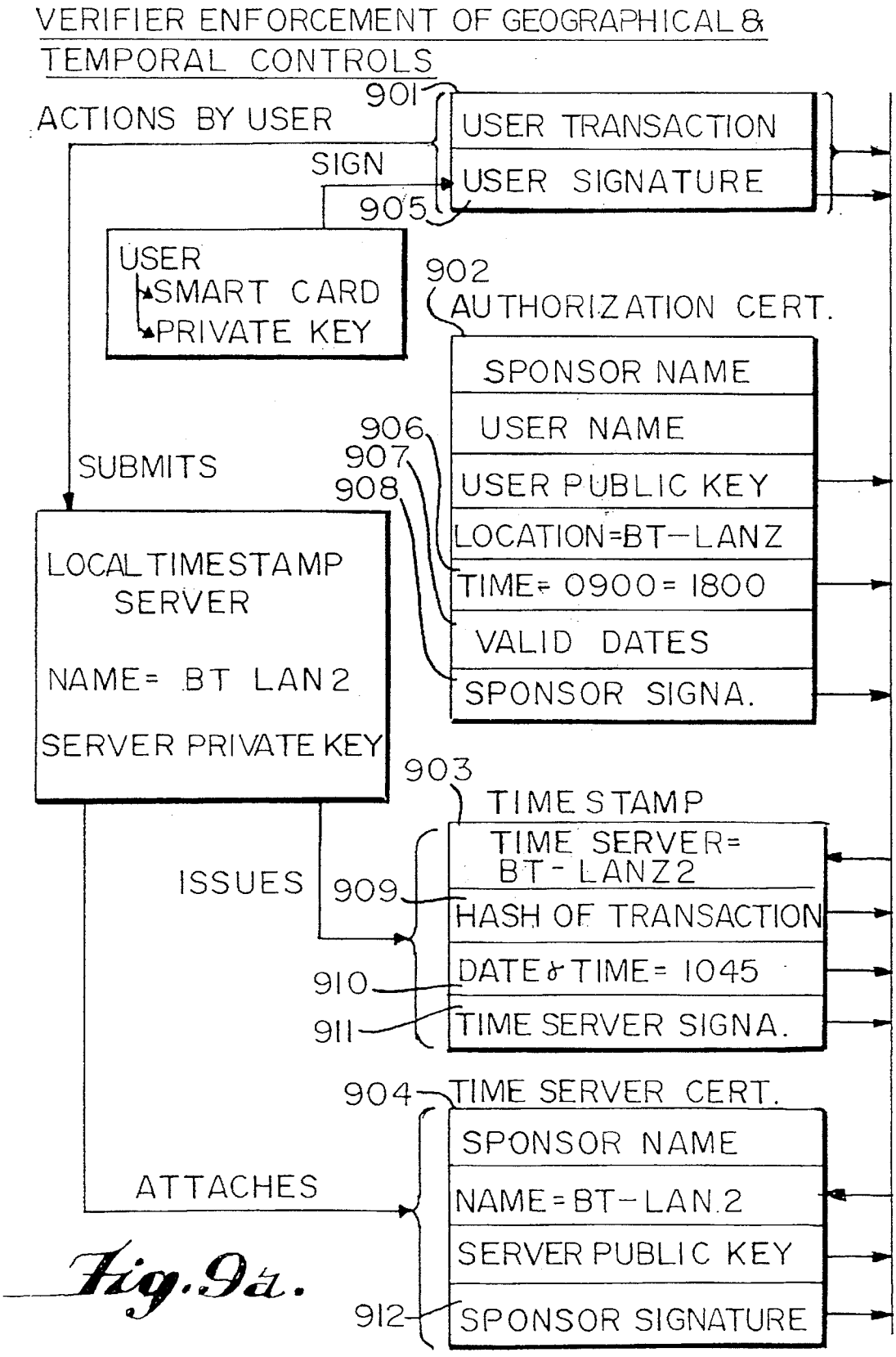
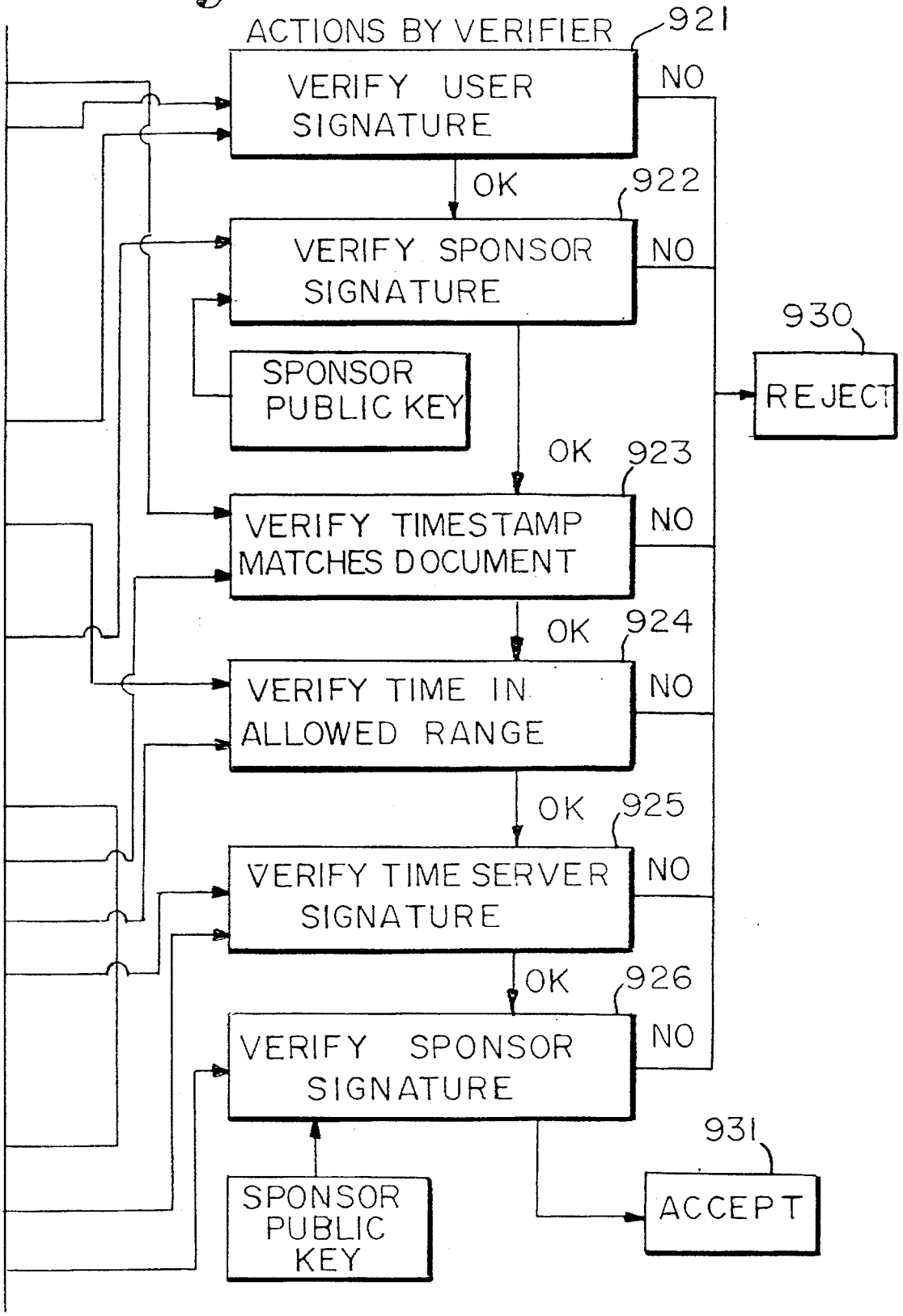

VERIFIER ENFORCEMENT OF GEOGRAPHICAL & TEMPORAL CONTROLS
ACTIONS BY USER
901
USER TRANSACTION
SIGN
USER SIGNATURE
905
USER
SMART CARD
PRIVATE KEY
902
AUTHORIZATION CERT.
SPONSOR NAME
USER NAME
906
907
908
SUBMITS
USER PUBLIC KEY
LOCATION=BT-LANZ
TIME= 0900= 1800
VALID DATES
SPONSOR SIGNA.
LOCAL TIMESTAMP SERVER
NAME= BT LAN2
SERVER PRIVATE KEY
903
TIMESTAMP
TIME SERVER= BT-LANZ2
ISSUES
909
HASH OF TRANSACTION
910
DATE & TIME= 1045
911
TIME SERVER SIGNA.
904
TIME SERVER CERT.
SPONSOR NAME
ATTACHES
NAME=BT-LAN.2
SERVER PUBLIC KEY
912
SPONSOR SIGNATURE

Fig. 9a.

Fig. 9b.


ACTIONS BY VERIFIER
921
VERIFY USER SIGNATURE
NO
OK
922
VERIFY SPONSOR SIGNATURE
NO
SPONSOR PUBLIC KEY
930
REJECT
OK
923
VERIFY TIMESTAMP MATCHES DOCUMENT
NO
OK
924
VERIFY TIME IN ALLOWED RANGE
NO
OK
925
VERIFY TIME SERVER SIGNATURE
NO
OK
926
VERIFY SPONSOR SIGNATURE
NO
931
SPONSOR PUBLIC KEY
ACCEPT

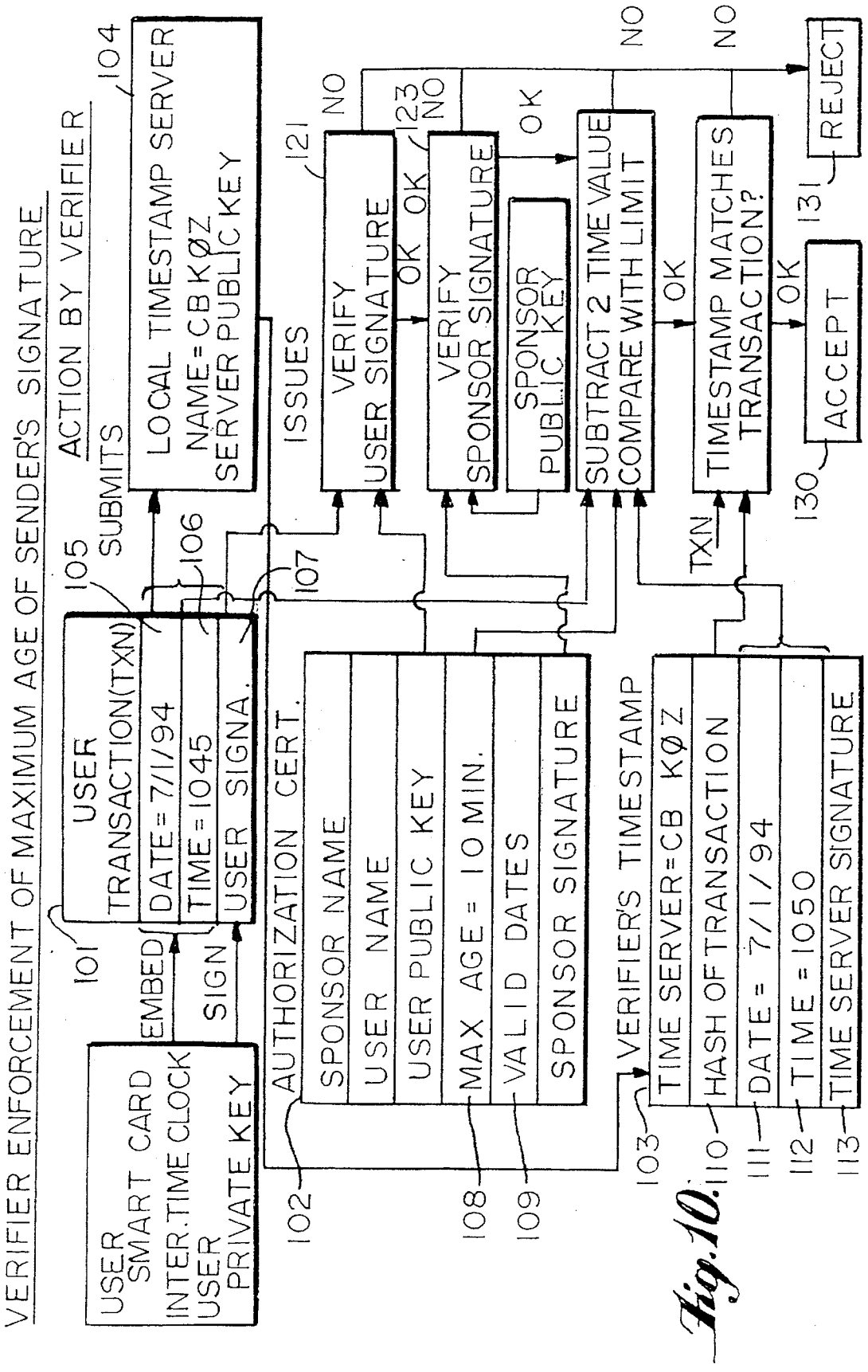

VERIFIER ENFORCEMENT OF MAXIMUM AGE OF SENDER'S SIGNATURE
ACTION BY VERIFIER
USER SMART CARD
INTER.TIME CLOCK
USER PRIVATE KEY
EMBED
SIGN
101
USER TRANSACTION(TXN)
DATE=7/1/94
TIME=1045
USER SIGNA.
105
SUBMITS
106
104
LOCAL TIMESTAMP SERVER
NAME=CBKØZ
SERVER PUBLIC KEY
107
ISSUES
AUTHORIZATION CERT.
SPONSOR NAME
USER NAME
USER PUBLIC KEY
MAX AGE = 10 MIN.
VALID DATES
SPONSOR SIGNATURE
102
108
109
121
VERIFY USER SIGNATURE
NO
OK
123
VERIFY SPONSOR SIGNATURE
NO
OK
SPONSOR PUBLIC KEY
SUBTRACT 2 TIME VALUE COMPARE WITH LIMIT
NO
OK
VERIFIER'S TIMESTAMP
TIME SERVER=CB KØZ
HASH OF TRANSACTION
DATE= 7/1/94
TIME=1050
TIME SERVER SIGNATURE
103
110
111
112
113
TXN
TIMESTAMP MATCHES TRANSACTION?
NO
OK
131
REJECT
130
ACCEPT

Fig. 10.

SPONSOR ENFORCEMENT OF PRE-APPROVED COUNTERPARTY RESTRICTION

Fig. 11a.

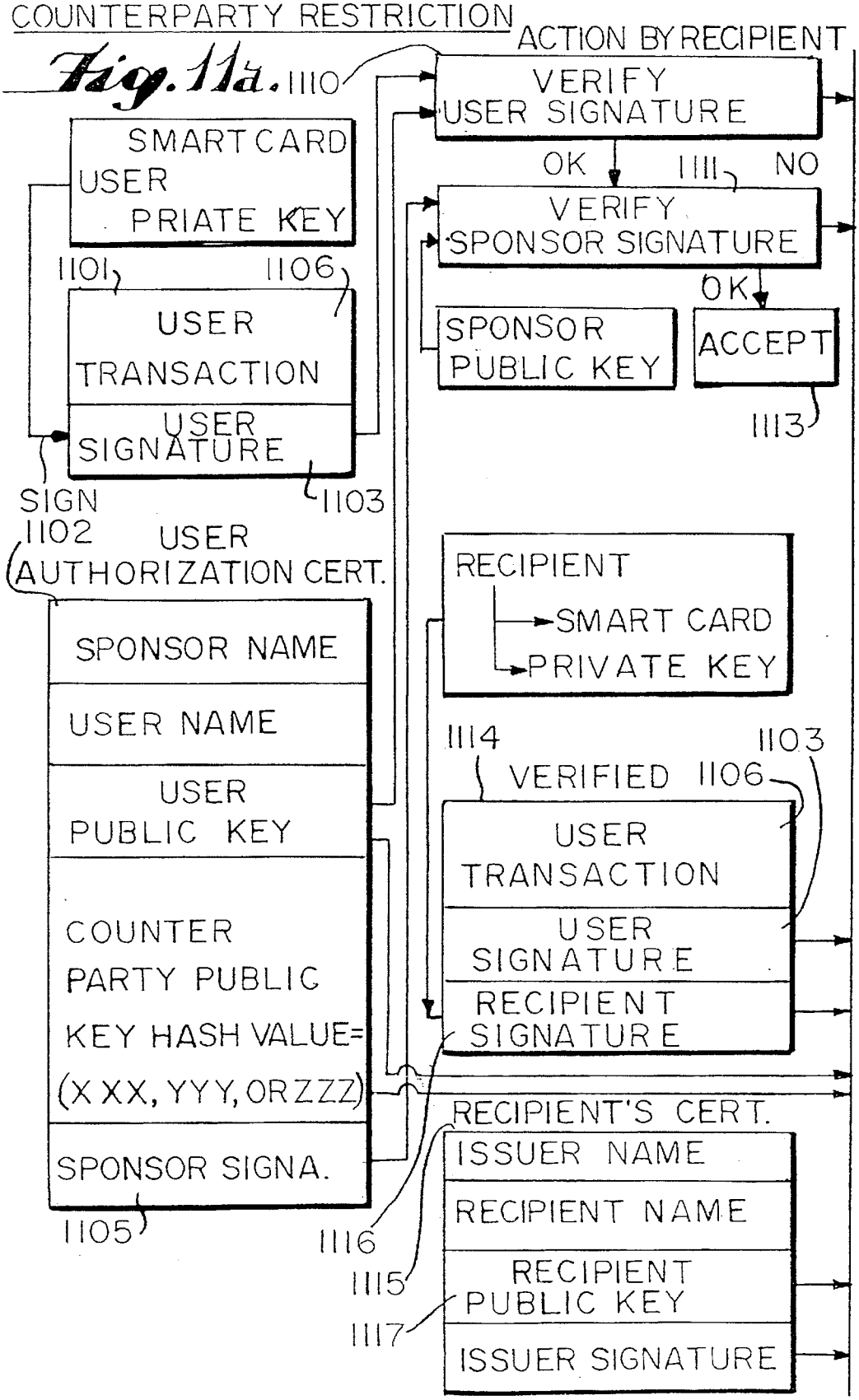
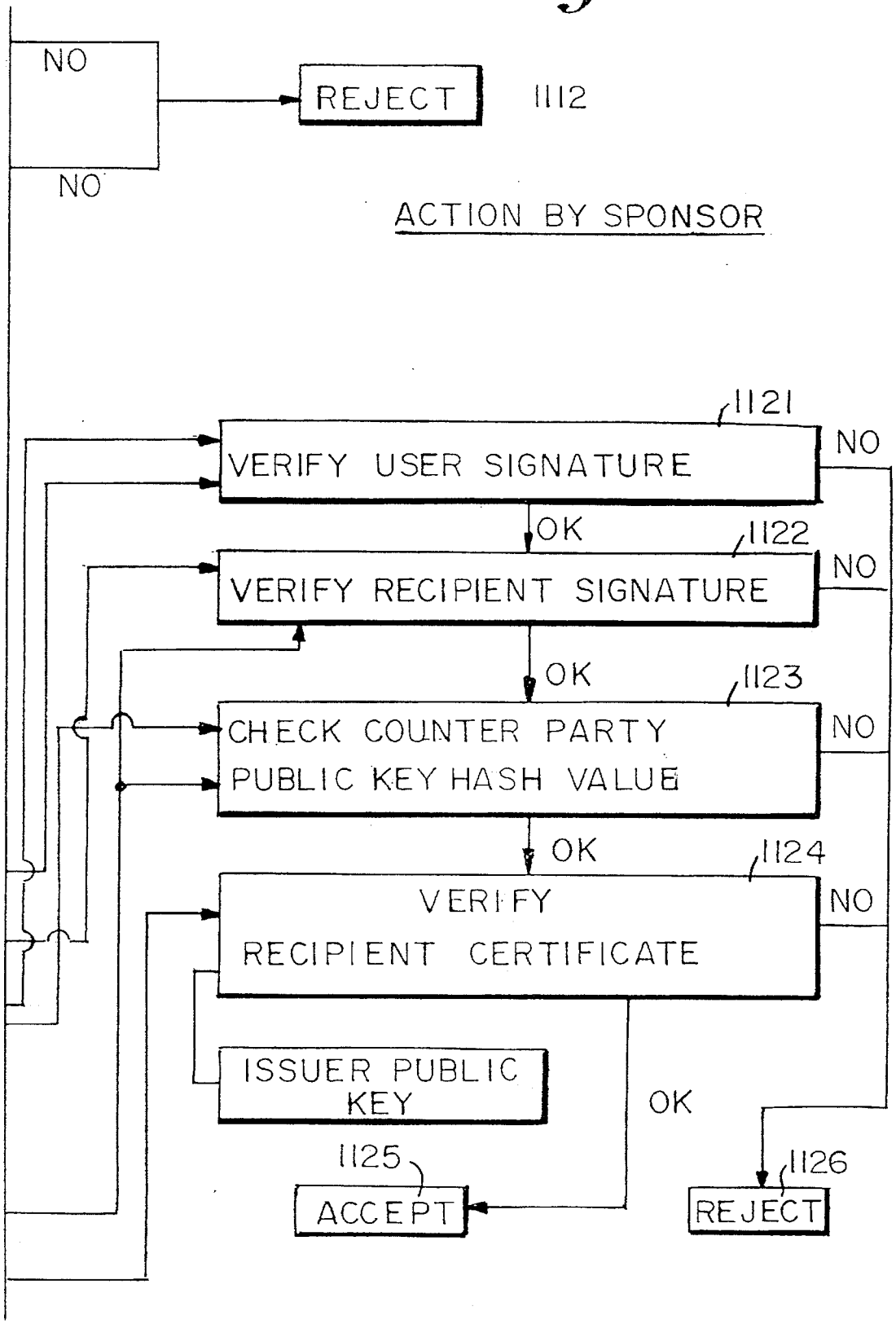


Fig. 11b.


NO
REJECT
1112
NO
ACTION BY SPONSOR
1121
VERIFY USER SIGNATURE
NO
OK
1122
VERIFY RECIPIENT SIGNATURE
NO
OK
1123
CHECK COUNTER PARTY PUBLIC KEY HASH VALUE
NO
OK
1124
VERIFY RECIPIENT CERTIFICATE
NO
ISSUER PUBLIC KEY
OK
1125
ACCEPT
1126
REJECT

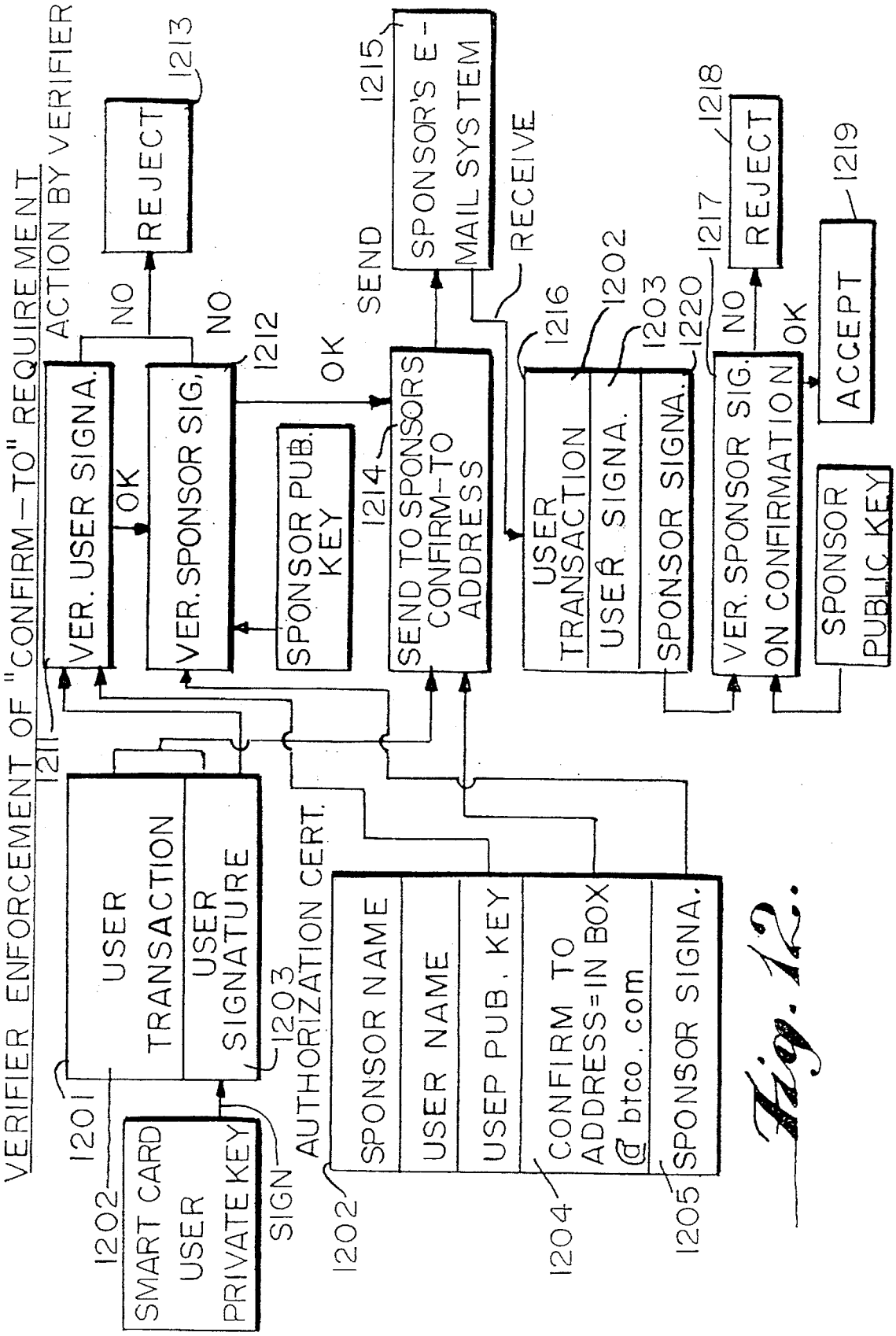

VERIFIER ENFORCEMENT OF "CONFIRM-TO" REQUIREMENT
ACTION BY VERIFIER
1211
VER. USER SIGNA.
OK
NO
REJECT
1213
VER. SPONSOR SIG,
NO
1212
OK
SPONSOR PUB. KEY
SEND
1215
SPONSOR'S E-MAIL SYSTEM
RECEIVE
1214
SEND TO SPONSORS CONFIRM-TO ADDRESS
1216
USER TRANSACTION
1202
USER SIGNA.
1203
SPONSOR SIGNA.
1220
NO
1217
1218
REJECT
VER. SPONSOR SIG. ON CONFIRMATION
OK
ACCEPT
1219
SPONSOR PUBLIC KEY
1201
USER TRANSACTION
USER SIGNATURE
1203
AUTHORIZATION CERT.
SIGN
1202
SMART CARD USER PRIVATE KEY
1202
SPONSOR NAME
USER NAME
USEP PUB. KEY
CONFIRM TO ADDRESS=IN BOX @ btco.com
1204
SPONSOR SIGNA.
1205

Fig. 12.

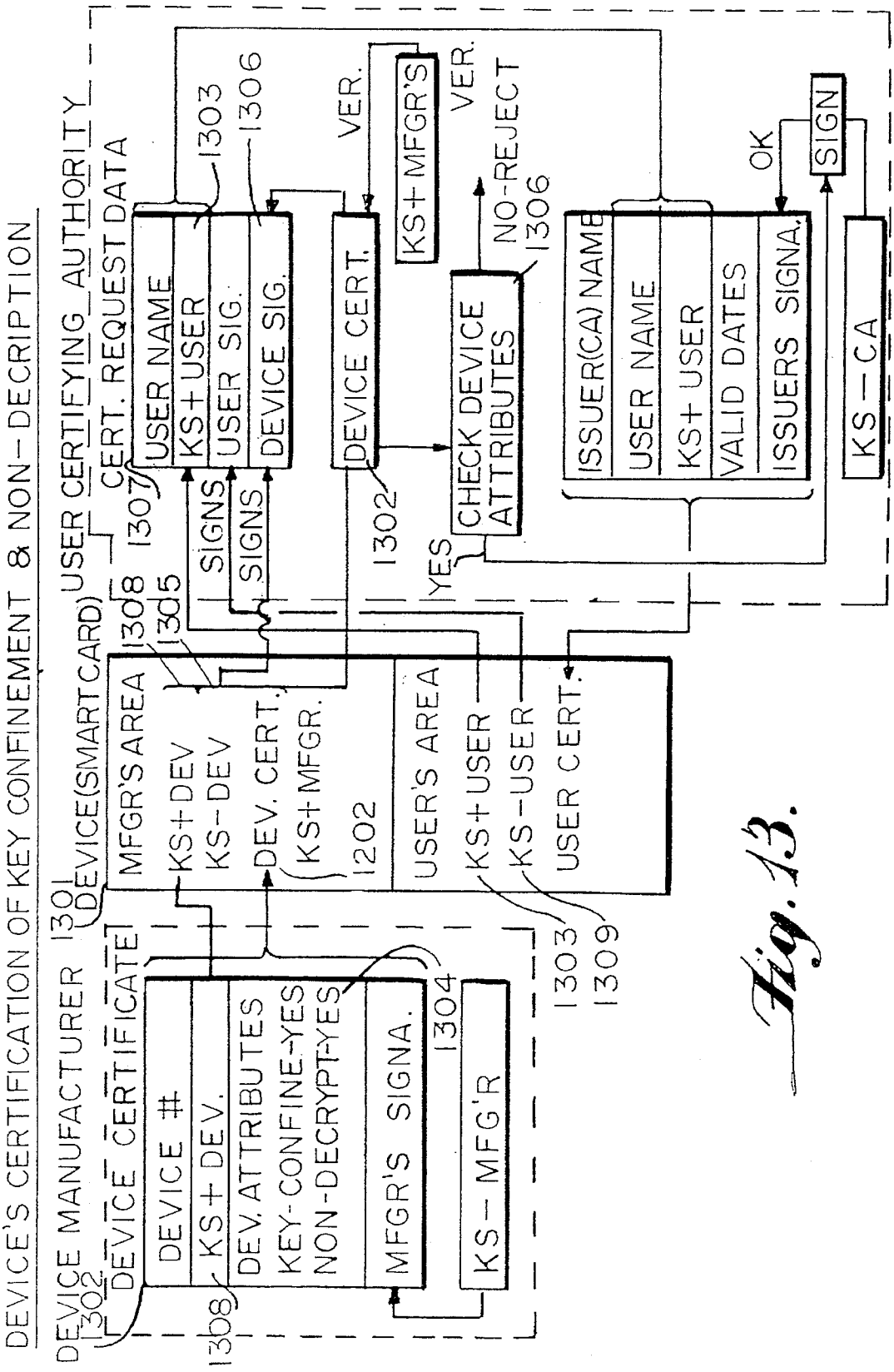

DEVICE'S CERTIFICATION OF KEY CONFINEMENT & NON-DECRIPTION
DEVICE MANUFACTURER 1301
DEVICE CERTIFICATE
1302
DEVICE #
KS+DEV.
DEV. ATTRIBUTES
KEY-CONFINE-YES
NON-DECRYPT-YES
MFGR'S SIGNA.
1304
KS-MFG'R
1308
DEVICE(SMARTCARD)
MFGR'S AREA
KS+DEV
KS-DEV
DEV. CERT.
KS+MFGR.
1202
USER'S AREA
KS+USER
KS-USER
USER CERT.
1303
1309
1308
1305
SIGNS
SIGNS
USER CERTIFYING AUTHORITY
CERT. REQUEST DATA
1307
USER NAME
KS+USER
USER SIG.
DEVICE SIG.
1303
1306
DEVICE CERT.
1302
VER.
KS+MFGR'S
VER.
CHECK DEVICE ATTRIBUTES
YES
NO-REJECT
1306
ISSUER(CA) NAME
USER NAME
KS+USER
VALID DATES
ISSUERS SIGNA.
OK
SIGN
KS-CA

Fig. 13.

METHOD FOR SECURELY USING DIGITAL SIGNATURES IN A COMMERCIAL CRYPTOGRAPHIC SYSTEM

This is a continuation of application Ser. No. 08/277,438, filed on Jul. 19, 1994, which was abandoned upon the filing hereof.

BACKGROUND OF THE INVENTION

This invention relates to digital signatures. More particularly, this invention relates to the use of digital signatures and certificates for digital signatures in a commercial cryptographic system for enforcing security policies and authorization requirements in a manner that reduces risks to the users.

Public-key cryptography is a modern computer security technology that can support the creation of paperless electronic document systems, providing that the user's digital signature on an electronic document, i.e., the user's electronic authentication and verification of the electronic document, can be given sufficient practical and legal meaning. Such paperless electronic document systems, or "document architectures," will encompass not only trading partners operating under standard bilateral contracts but also global multilateral systems in which any entity can, in theory, correspond with any other entity in a legally provable manner, assuming that proper security controls are observed throughout.

These systems will have enormous commercial significance because, in many cases, cost reductions on the order of 10-to-1 can be realized over current paper transaction procedures. This improvement is sufficiently dramatic such that many organizations would, for economic and competitive reasons, be compelled to use them once their practicality had been demonstrated.

No one disputes that paper is a bothersome anachronism in the electronic world or that verifying pen-and-ink signatures is costly and error-prone. At least with paper, however, the signer retains the basic "contextual controls" of document preparation and physical delivery. On a digitally signed electronic document, on the other hand, a signer controls only the encoded signature. All time, place and manner controls are absent, and nothing distinguishes a valid user signature from one fraudulently produced by another user who somehow obtained the first user's smart card and PIN. It would not take too many multi-million or multi-billion dollar losses to erase all the savings produced by this "newfangled" office-automation technology. Therefore, digital signatures will see early use only in consumer "electronic coin purse" applications, where exposure is low, and in wholesale financial transfers, as to which extremely tight security procedures are already the norm. However, these uses will have little general commercial impact.

Thus far, major corporations and banks have declined to invest in these technologies due to lack of well-defined risk models and auditing standards and due to uncertainties regarding legal and liability issues. Serious investments to commercialize digital signatures will occur only after leading national auditing and legal experts have ruled that these systems contain adequate security controls to warrant reliance in mainstream intra- and inter-corporate business transactions, typically in the $10,000 to $10 million range. In order for this goal to be achieved, security controls must be formulated to reduce the risks of participants in digital signature document systems to the absolute lowest level technically achievable.

There are two types of cryptographic systems in which digital signatures have been used: symmetric and asymmetric cryptosystems. FIG. **1** illustrates the use of symmetric and asymmetric algorithms for encryption. In symmetric (conventional) cryptography, as shown in FIG. **1**(*a*), the sender and recipient of a communication share a secret key **11**. This key is used by the sender, the originator of a communication, to encrypt the message **12** and by the recipient of the communication to decrypt the message **13**. It may also be used by the recipient to authenticate a message by having the sender use the secret key to compute some function such as a Message Authentication Code (MAC) based upon the message; the recipient thus can be assured of the identity of the originator, because only the sender and the recipient know the secret key used to compute the MAC. DES is an example of a symmetric cryptosystem.

In asymmetric (public key) cryptography, shown in FIG. **1**(*b*), different keys are used to encrypt and decrypt a message. Each user is associated with a pair of keys. One key **15** (the public key) is publicly known and is used to encrypt messages **17** destined for that user, and the other key **16** (the private key) is known only to that user and is used to decrypt incoming messages **18**. Since the public key need not be kept secret, it is no longer necessary to secretly convey a shared encryption key between communicating parties prior to exchanging confidential traffic or authenticating messages. RSA is the most well-known asymmetric algorithm.

A digital signature, however, is a block of data appended to a message data unit, and allows the recipient to prove the origin of the message data unit and to protect it against forgery. Some asymmetric algorithms (e.g., RSA) can also provide authentication and non-repudiation through use of digital signatures. In order to sign data, the sender encrypts the data under his own private key. In order to validate the data, the recipient decrypts it with the sender's public key. If the message is successfully decrypted using the sender's public key, the message must originally have been encrypted by the sender, because the sender is the only entity that knows the corresponding private key. Using this method of signing documents, the encrypted message is bound to the signature, because the recipient cannot verify the message without decrypting the signature data block. The signature-encrypted message can then be encrypted to the recipient using the recipient's public key, as usual.

Digital signatures may also be formed using asymmetric encryption algorithms as described below and as illustrated in FIG. **2**. To sign a message, the message **20** is first digested (hashed) into a single block **22** using a one-way hash function **21**. A one-way hash function has the property that, given the digest, it is computationally infeasible to construct any message that hashes to that value or to find two messages that hash to the same digest. The digest **22** is then encrypted with the user's private key **23**, and the result **24** is appended to the encrypted or unencrypted message as its signature **25**. The recipient uses the sender's public key **26** to decrypt the signature **25** into the hash digest **22**. The recipient also digests (hashes) the message **20**, which has been received either unencrypted or encrypted and then decrypted by the recipient, into a block **27** using the same one-way hash function **21** used by the sender. The recipient then verifies **28** the sender's signature by checking that the decrypted hash digest **22** is the same as the hashed message digest **27**.

Separating the signature from the message in this way, i.e., not requiring the sender and recipient to encrypt and

decrypt the entire message in order to verify the signature, greatly reduces the amount of data to be encrypted. This is important because public key algorithms are generally substantially slower than conventional algorithms, and processing the entire message in order to verify a signature would require a significant amount of time. The signature process also introduces redundancy into the message, which, because the message must hash to the specified digest, allows the recipient to detect unauthorized changes to the message.

A digital signature provides the security services of (a) integrity, because any modification of the data being signed will result in a different digest and thus a different signature; (b) origin authentication, because only the holder of the private key corresponding to the public key used for validation of the signature could have signed the message; and (c) non-repudiation, as irrevocable proof to a third party that only the signer, and not the recipient or its employees, could have created the signature. A symmetric secret key authenticator, e.g. the X9.9 FLAC, does not provide these services, since either of the two parties can create the authenticator using their shared key.

Several of the mechanisms discussed herein assume the ability to attach multiple signatures or cosignatures to a document. RSA Data Security, Inc., "PKCS #7: Cryptographic Message Syntax," 1993, which is hereby incorporated by reference, defines a useful format for this purpose. Each signature structure on a document will contain an indication of the certificate needed to validate the signature along with a bit string containing the actual signature. Additionally, other information relevant to the particular signer may be included in an individual signature computation. This per-signer information may be included in the signature computation as "signature attributes."

In order for one user to identify another user for transmission of a message in a way that ensures the second user's possession of a private key, the first user must be able to obtain the other user's public key from a trusted source. A framework for the use of public key certificates was defined in CCITT, "X.509: The Directory: Authentication Framework," April, 1993 ("X.509"), which is hereby incorporated by reference. These basic public key certificates bind a user's name to a public key and are signed by a trusted issuer called a Certification Authority (CA). Besides containing the user's name and public key, the certificate also contains the issuing CA's name, a serial number and a validity period.

Although X.509 does not impose any particular structure on the CAs, many implementations find it reasonable to impose a hierarchical structure in which each CA (in general) certifies only entities that are subordinate to it. Hence, we can construct a hierarchy of Cas, as shown in FIG. **3**, in which the higher level CAs **31** (perhaps banks) sign the certificates **34** of the CAs **32** beneath them (e.g., companies), and the lowest level of CAs **32** sign user **33** certificates **35**. At the top of this hierarchy (not shown) are a relatively few other root CAs, perhaps one per country, that may "cross-certify" each other's public keys (root keys).

Various security architectures define mechanisms to construct a certification path through the hierarchy to obtain a given user's certificate and all CA certificates necessary to validate it. These architectures share the common characteristic that a user need trust only one other public key in order to obtain and validate any other certificate. The trusted key may be that of the top-level CA (in a centralized trust model) or of the local CA that issued the user's certificate (in a decentralized model).

Certificates also contain an expiration date. If it is necessary to cancel a certificate prior to its expiration date, such as if the name association becomes invalid or the corresponding private key is lost or compromised, the certificate may be added to the CA's certificate revocation list (CRL) or "hot list." This list is signed by the CA and widely distributed, possibly as part of the CA's directory entry. The certificate remains on the CRL until the certificate's expiration date.

Often certain information concerning an entity or CA needs to be made available in a trusted manner. In a secure X.500 Directory, this information would be retrieved via standard Directory operations and the result would be signed by the Directory. In the absence of such a secure X.500 implementation, this information is placed in an attribute certificate, which is signed by a CA in the same manner as the public key certificate. Attribute certificates would be created on presentation of the proper credentials by the user. For example, the user would present his public key certificate and prove he possesses the corresponding private key, as one form of identification. Attribute certificates are linked to the user's basic public key certificate by referencing the basic certificate's serial number and are revoked by an identical parallel CRL mechanism. Attribute certificates are discussed further in ANSI X9F1, "X9.30 Part 3: Certificate Management for DSA," June, 1994, and U.S. Pat. Nos. 4,868,877, 5,005,200 and 5,214,702, which are all hereby incorporated by reference.

An attribute certificate is a structure separate from a public key certificate because proper separation of duties may often require that the CA that issues the attribute certificate be different than the CA that issues the public key certificate. A central CA might rarely of itself possess the required security or authority to "sign for" all of a user's authorizations. Having separate CAs generate various types of attribute certificates distributes risks more appropriately. In addition, the defined attributes may not be required for all domains, networks or applications. The need for these attributes and for additional domain-specific attributes is determined by each domain.

The user's basic public key certificate remains X.509 compatible, allowing its use with other applications and allowing use of commercial products for certificate generation.

It is desirable to be able to construct a trusted organization that utilizes digital signature and certificate mechanisms to enforce a security policy defined by rules within this organizational structure.

It is also desirable to use digital signature and certificate mechanisms to encode industry-wide security policy and authorization information into the signatures and certificates in order to permit the verifier of a signature to decide whether to accept the signature or certificate as valid, thus accommodating and easing electronic commerce business transactions.

It is further desirable to reduce the risks associated with digital signature systems, particularly with end-user smart cards, by building on this use of public key certificates and attribute certificates.

It is further desirable to prevent the use of such a digital signature system by any party that might purport to "accept" a transaction in contravention of the applicable authorization certificates when that party had not signed the applicable "system rules" agreement pertaining to that system of communicating signer authorization.

SUMMARY OF THE INVENTION

These and other objects of the invention are accomplished in accordance with the principles of the invention by pro-

viding a system for securely using digital signatures in a commercial cryptographic system that allows industry-wide security policy and authorization information to be encoded into the signatures and certificates by employing attribute certificates to enforce policy and authorization requirements. In addition to value limits, cosignature requirements and document type restrictions that can be placed on transactions, an organization can enforce with respect to any transaction geographical and temporal controls, age-of-signature limitations, preapproved counterparty limitations and confirm-to requirements by using attribute certificates for the transacting user. Restrictions on distribution of certificates can be set using attribute certificates. Certificates can be used also to ensure key confinement and non-decryption requirements of smartcards in this system.

BRIEF DESCRIPTION OF THE DRAWINGS

The above and other objects and advantages of the invention will be apparent upon consideration of the following detailed description, taken in conjunction with the accompanying drawings, in which the reference characters refer to like parts throughout and in which:

FIG. **1** is a diagram showing the prior art use of symmetric and asymmetric algorithms for encryption;

FIG. **2** is a flow chart illustrating the prior art process of a digital signature using an asymmetric encryption algorithm;

FIG. **3** is a diagram showing a hierarchy of signature certification authorities;

FIG. **4** is a diagram showing a directory information tree (DIT);

FIG. **5** is a diagram showing an example of an authorization certificate;

FIG. **6** is a flow chart illustrating the prior art process of verifier enforcement of a transaction monetary value restriction;

FIG. **7** is a flow chart illustrating the prior art process of verifier enforcement of a transaction cosignature requirement;

FIG. **8** is a flow chart illustrating the process of verifier enforcement of a transaction document-type restriction;

FIG. **9** is a flow chart illustrating the process of verifier enforcement of a transaction geographical and temporal control;

FIG. **10** is a flow chart illustrating the process of verifier enforcement of a maximum age of sender's signature restriction;

FIG. **11** is a flow chart illustrating the process of verifier and sponsor enforcement of a pre-approved counterparty restriction;

FIG. **12** is a flow chart illustrating the process of verifier enforcement of a transaction "confirm-to" requirement; and

FIG. **13** is a flow chart illustrating the process of a device's certification of key confinement and nondecryption.

DETAILED DESCRIPTION OF THE INVENTION

The following general principles and philosophies are reflected in the signature verification model defined in this invention. First, CA and user certificates can contain attributes that document the conditions and assumptions under which they were created. Verifiers may simply reject all certificates and transactions that do not meet their minimum standards.

Also, attribute certificates may be signed by a user's "sponsor" to signify that the sponsor's signature will be honored for official business if the transaction meets the requirements stated or implied by the attributes. Although typically the user's sponsor will be the user's employer, the model can be extended to include the user's bank, credit card issuer, voting bureau, video rental store, public library or any other entity that might accept the user's signature. This sponsor (authorization) certificate is thus the electronic equivalent of an "affidavit of legal mark," as used in the context of a traditional signature stamp. See Robert Jueneman, "Limiting the Liability of CAs and Individuals Regarding the Use of Digital Signatures," presented to the ABA Section of Science and Technology Certification Authority Work Group, Jul. 2, 1993.

Furthermore, industries may develop "industry policy" statements that establish minimum requirements for signature verification. All participants would sign these multilateral agreements in order to ensure that all counterparties would be bound by the encoded restrictions. Normally, sponsor certificates should be required in all cases, and digital signatures would be deemed otherwise null and void in their absence. Industry-wide policies would also define (1) relevant document types and classes, (2) signer roles and titles, and (3) coded symbols for incorporating by reference standard contractual terms and conditions.

Moreover, there must be strict adherence to the principle that all restrictions can be enforced in an entirely automated manner (i.e., verification "on sight"), without reference to paper agreements or human interpretation, sometimes also termed "fully machineable straight-through processing." In complex and/or high-volume environments, this is required in order to give these security controls credibility in the eyes of audit and legal experts. Reference to trusted third parties should also be minimized to reduce verification latency times.

While these restrictions seem complex, they merely reflect ordinary business procedures made explicit for purposes of machine verification. Formerly, such controls were enforced inside the sponsor's computer systems before sending out the transaction. However, with the advent of multilateral distributed transactions, the verifying user is typically off-line from his sponsor's system, and so the verifier must enforce the sponsor's authorization model, as reflected in the attribute certificates. Once this methodology is specified, office software vendors will develop menu-driven systems to create and manage user attributes, and the cost to user organizations will be relatively low.

Organizational Structure in Certificates

The certificates themselves may reflect the structure of a sponsor organization. Because many authorization decisions are based on the user's position in an organization, the organizational structure and the user's position therein may be specified as part of a user's name. Names in certificates are specified in terms of the X.500 Directory model, as follows.

The X.500 Directory structure is hierarchical; the resulting distributed database comprises the Directory Information Tree (DIT), as shown in FIG. **4**. Each entry **41** is of a specific object class and consists of a set of properties called attributes **42**. An attribute **42** consists of a type **43** and one or more values **44**. Thus, in an entry of class organization, one attribute is the organizationName; in an entry of class organizationalPerson, attributes might include title and telephoneNumber.

Each entry also has one or more special attribute values used to construct the object's name; this attribute value is the relative distinguished name (RDN) of the entry. An object's distinguished name (DN) **45**, which is created by concatenating the relative distinguished names **46** of all entries from the DIT root to the entry, uniquely identifies the object in the global DIT.

Several of the attributes defined in X.500 may be usefully included in the user's attribute certificate. For example, the object class can be used to distinguish between entities (e.g. users and roles) whose distinguished names are of the same form. Also, the title may be used in making authorization decisions.

In addition to the use of the DIT to group entities along organizational lines, X.500 defines several object classes that can be used to construct arbitrary groups of entities. These object classes include the organizational role, whose "role occupant" attribute lists the names of the users who occupy the role, and the group of names, whose "member" attribute lists the names of group members. To convey this information in a trusted way, one could define role and group certificates that convey the names of the role occupants or group members, respectively, and that are signed by a CA, thus enabling use of this feature outside the context of an X.500 directory system.

Group and role certificates may be used in conjunction with a cosignature mechanism to simplify the construction of cosignature requirements. For example, a transaction might require the signatures of three occupants of the "purchasing agent" role. A user may also indicate the role in which he is acting by including the role in the signature computation as a (per-signer) signature attribute. The asserted role may then be matched against a role certificate (or the user's attribute certificate) during verification.

Policy Information in Certificates

It is another embodiment of this invention to encode information regarding a CA's security policy into the attribute certificates of the CA and its subscribers, so that the verifier of a signature can use the information in determining whether to accept a signature as valid. In general, the CA's certificate will convey the rules that a CA uses when making certification decisions, while the user's certificate will convey the information used by the CA when applying these rules.

Attributes in CA certificates can indicate security policy and assurance information for a particular CA. This policy information can also be inherited by subordinate CAs, allowing easy construction of security domains sharing a common policy. Policy attributes in a CA's certificate might, among others, include:

(1) Liability Limitations: the extent to which a CA is liable in the event of various problems (e.g., CA key compromise, defective binding); this might be no liability, full liability or a specific monetary amount.

(2) Trust Specification: a description of which users and CAs a given CA can certify, expressed relative to the CA itself (e.g., "all subordinates"), or to the DIT in general (e.g., "the subtree below Organization ABC"), or to others.

(3) Required Attributes: a list of those attributes in the user's attribute certificates that must be verified against a transaction and/or its context in order for the transaction to be considered authorized. These attributes would be found in the certificate(s) of the sponsor and allow a single authorization certificate to contain authorization attributes for use with multiple applications. Some suggested user authorization attributes are defined later.

(4) Allowable Name Forms: a specification of the allowable name forms that the CA may certify. This information is held as (a) a set of name bindings, which defines the attributes that may be used to name entries of a given object class (i.e., the allowable RDN formats for entries of that class), and (b) a set of structure rules, which defines which object classes may be adjacent (i.e. superior or subordinate) to each other in the DIT, i.e., the order in which object classes may be chained together to form a complete DN. This policy attribute may be used to restrict the type of entities that may sign transactions. For example, for wire transfer applications, it might be desirable to restrict signature capability to the organization itself, rather than to users within the organization, since this is similar to the current mode of operation using DES MACs.

(5) Cross-Certificates: it may be desirable from an efficiency point of view to allow certifying entities and as organizations to cross-certify each other in order to constrain the length of certification paths. On the other hand, it is not desirable to allow certification paths to contain arbitrary numbers of cross certificates, as it is difficult to determine the level of trust in the entity at the other end. Many certification architectures restrict certification paths to contain only one cross-certificate. To accommodate a wider range of policies, an attribute may be added to the attribute certificate associated with the cross-certificate indicating that the cross-certifier explicitly allows the use of cross-certificates issued by the CA being cross-certified.

Attributes in a user's or entity's attribute certificate may represent the information verified by the CA when creating the certificate for the entity. Policy attributes in a user's certificate might, among others, include:

(1) Binding Information: the criteria used to bind the public key to the identity of the entity being certified. This includes (a) the method of delivery, such as being presented in person, by authorized agent, by mail or by another method; (b) the method of identification, such as by reasonable commercial practices, verified by trusted third party, dual control, fingerprint check, full background investigation or another method; (c) the identification documents presented to the CA; and (d) the subject's entity type, i.e. individual, corporation or other.

(2) Trusted Third Parties: the names of any trusted third parties or agents involved in the binding process.

(3) Roles: it may be useful for authorization purposes to indicate which roles (both internal and external to the organization) a user may exercise. This is in contrast to a role certificate, which would be issued to the role and contain the names of all occupants.

(4) Relative Identity: a CA may wish to certify only a portion of the DN of an individual. In particular, the CA might disclaim liability for correctness of an individual's personal name, since, under legal Agency principles, the individual's signature is binding on their organizational sponsor in any event. Consider the name:

C=US; O=Bankers Trust; OU=Global Electronic Commerce; CN=Frank Sudia; TI=VP

The CA might certify only the validity of the organization, organizational unit and title portions of the individual's distinguished name, all of which are easy to verify, while the personal name would only be "reasonably believed accurate." In view of the relative ease of obtaining false identity papers, this avoids the need for prohibitively expensive background investigations. Such an identification can be relied on in an ordinary commercial setting but not in a proceeding concerning a will or inheritance, for example.

(5) Absolute Identity: we define relative identity as the user's identity "relative" to his organizational sponsor. Put another way, we certify all elements of the user's "business card identity," except his personal name. As a special case, some CAs might undertake to certify the absolute identity of selected users, say the children of wealthy clients, diplomats or national security operatives, almost certainly bolstered with biometric techniques. This would be rare and is presented here only for completeness in order to round out the "relative identity" concept.

Authorization Information in Certificates

Attributes may convey restrictions that control the conditions under which a signature is valid. Without such restrictions, the risk of forgery would be considered excessive, since an electronic signature can be affixed to almost any digital document by anyone possessing the user's smart card and personal identification number (PIN). In the electronic environment, the normal contextual controls of document creation and physical delivery are either weak or nonexistent.

Even authentic users are hardly trustworthy to undertake free-form offline commitments, and organizations will thus welcome the capability to positively restrict the scope of express signature authorization. Such authorization attributes might, in addition to standard X.500 attributes, include Transaction Limits, Cosignature Requirements, Document Types, Authorized Signatories, Geographical and Temporal Controls, Age of Signature, Preapproved Counterparties, Delegation Controls, and Confirm-To Requirement. These attributes can be encoded in one or more authorization certificates signed by the signer's organizational sponsor or by an external CA acting on behalf of the organization. An example of an authorization certificate and an associated transaction is shown in FIG. **5**.

When a recipient user (verifier) receives a transaction **51** from a sending user, the recipient first uses the sender's basic key certificate **55** to verify the sender's signature **52** on the transaction **51**. As will be described in greater detail below, the recipient also uses the sender's authorization certificate **56**, signed by the sender's sponsor **59**, to verify the cosignatures **53** and timestamp notarization **54** appended to the transaction **51** and to verify that the attribute values **57** of the transaction **51** fall within the authorized attribute values **58** as specified in the authorization certificate **56**.

The user may be subject to transaction limits that control the value of transactions or other documents that the user may initiate. The user's signature will be valid only on transactions originated either up to a certain monetary limit or between two monetary value boundaries. Accordingly, as shown in FIG. **6**, the sending user sends a transaction **601** signed **603** by the sender (actually by the user's smart card **600** containing his private key) and appends thereto an authorization certificate **604**. The verifier uses the authorization certificate **604** to verify **607** the user's signature **603** and to verify that the transaction monetary value **602** falls within the transaction limit attribute value **605** in the authorization certificate **604**. The verifier also verifies **609** the sponsor signature **606** on the authorization certificate **604** using the sponsor's public key **610**. If any of these signatures and attribute values does not verify, the transaction is rejected **611**. If verification is complete, the transaction is accepted **612**.

With regard to cosignature requirements, additional signatures may be required in order for a given signature to be considered valid. Quorum and weighting mechanisms can be used to construct fairly elaborate checks and balances for explicitly governing the level of trust in each user. The particular sequence or order of required signatures may also be specified. Referring to FIG. **7**, sending user A sends a transaction **702** signed **703** by his own smartcard **700** and, if user B's cosignature is required on the transaction **702**, signed **704** by the smartcard of user B **701**. Sending user A also appends his own authorization certificate **705** to the transaction **702**. The verifier uses the authorization certificate **705** to verify **711** user A's signature **703**, and uses the sponsor's public key **713** to verify **712** the sponsor's signature **707** on the authorization certificate **705**; if either signature does not verify, the transaction is rejected **720**. If a cosignature value **706** is required **714** by the authorization certificate **705**, the recipient enforces the requirement by verifying **715** cosigner user B's signature **704** on the transaction **702**, and then checks cosigner user B's public key certificate **708** by verifying **716** the signature **709** of the certificate issuer, using the issuer's public key **717**. If the signature of either user B or his certificate's issuer does not verify, the transaction is rejected **722**.

The use of cosignatures allows an organization to effectively define checks and balances, and to explicitly specify the level of trust in a user. The use of cosignatures also greatly reduces the risks that result from inadvertent compromise of a private key due to theft, misuse or misplacement of a smartcard or PIN. In particular, it is believed that the ability to require cosignatures, value limits and related controls will enable organizations to carefully manage and fine-tune all signature authorizations, thereby giving them all the tools needed to manage and limit their risks. Use of cosignatures further allows distribution of the authorization function over multiple locations and hardware platforms, with the resultant minimization of risks that might result from access control failures on one of those platforms. See U.S. Pat. Nos. 4,868,877, 5,005,200 and 5,214,702.

Authorization signatures, which must meet the restrictions specified in the signer's certificate, can also be distinguished from other cosignatures by including the signature purpose as a signature attribute and by requiring that an indication of the signature purpose be included in the data being signed. This signature-purpose attribute might require the values of: (a) an authorization signature appropriate to the document, (b) an authorization cosignature appropriate to the document, where the cosigner's certificate has sufficient authority to authorize the document, and (c) a witness cosignature, where the cosigner's certificate does not by itself have sufficient authority to authorize the document. Signature purpose encodings are discussed in draft ANSI standard X12.58 Version 2 (Appendix) issued by the Data Interchange Standards Association (DISA), which is hereby incorporated by reference.

The user can also be restricted to signing only particular document types, such as ordinary correspondence, purchase orders, specified EDI transaction types, business contracts, specified financial instruments, etc., as defined by industry-wide policies. It may also be desirable for efficiency to exclude certain large classes of transactions and documents. Referring to FIG. **8**, the recipient enforces the document-type restriction in the sender's transaction **801** by first verifying **807** the sender's signature **803** on the transaction and by then verifying **808** the document type attribute value **802** within the transaction **801** to enforce the document type restriction **805** within the sender's authorization certificate **804**. The recipient then verifies the authorization certificate **804** by using the sponsor's public key **811** to verify **809** the sponsor's signature **806**. If either a signature or the attribute restriction does not verify, the transaction is rejected **810**.

An organization can indicate that there are specific authorized signatories, that is, that only specific individuals can "sign for" the organization, similar to a standard "corporate resolution" to this effect. This might complement the document-type concept, as an additional control on signing of "corporate" document-types. This restriction can be implemented by specifying that a cosignature is required in which the cosigner's title (in its distinguished name) must be equal to one on a specified list contained in a authorization certificate. This is in lieu of naming a list of one or more required cosigners.

Geographical and temporal controls include locations and time periods from which transactions are considered valid. Use of a local trusted "timestamp notary" is assumed. Such a notary would append a trusted timestamp to the originator's signature on a document and would then sign the result. Thus, time-of-day and day-of-week restrictions would normally coincide with the work-week of the user's locale. Also, location information would be associated with the notary so as to restrict access to a specific network segment, typically the user's assigned work area. The "granularity" of location controls would depend on the network architecture. The signer or the signer's computer system must attach a certified timestamp from a specified local server to the transaction, or else the verifier cannot accept the transaction and the signer's sponsor will not be bound by it. As shown in FIG. **9**, the sending user attaches to the transaction **901** an authorization certificate **902**, as usual, an authorized timestamp **903** and a time server certificate **904**. The recipient verifies **921** the sender's signature **905** on the transaction **901** and verifies **922** the sponsor's signature **908** on the authorization certificate **902**. The recipient then (1) verifies **923** that the timestamp transaction text hash **909** matches the result of the text of the transaction **901** hashed with a known hash function, (2) verifies **924** that the time and date **910** on the transaction timestamp **903** fall within the authorized time and date **906** attribute values as specified in the authorization certificate **902**, (3) verifies **925** the time server signature **911** on the timestamp **903**, and (4) verifies **926** the sponsor's signature **912** on the time server certificate. If all these conditions are satisfied, the transaction is accepted **931**; if not, the transaction is rejected **930**.

Furthermore, a document may not be valid unless the signature is verified within some specified time period. For high-value transactions this age-of-signature attribute period would be quite short, while for more normal transactions, especially those sent via store-and-forward systems such as X.400, a longer interval (such as two days) would be appropriate. FIG. **10** shows enforcement by a recipient of the age-of-signature attribute value. The time of verification would be provided using a receipt **103** signed by a trusted timestamp service **104** containing, at a minimum, the recipient's name and the signature from the original transaction. The verifier must submit a timestamped copy of the original signature that is dated promptly after the time and date of the original transaction, or else the sponsor will reject it. As shown in FIG. **10**, the recipient (verifier) verifies **121** the sender's signature **107** on the transaction **101** and verifies **122** the sponsor's signature **115** on the authorization certificate **102**. The recipient then verifies **123** that the difference between the date **105** and time **106** on the transaction **101** and the date **111** and time **112** on the timestamp **103** is within the age-of-signature attribute restriction **108** in the authorization certificate **102**. The recipient also verifies **124** that the hash **110** of the transaction **101** within the trusted timestamp **103** matches the text of the transaction **101**. If all these conditions are satisfied, the transaction is accepted **130**; if not, the transaction is rejected **131**.

A "preapproved counterparties" attribute value restricts an entity to dealing only with some specified set of known trustworthy partners. This is a common requirement in dial-up home banking systems, which typically require that all authorized payees be specified in advance. Another way of stating this is that "free-form transfers" are forbidden. Sponsors realize that, in case of an error, they stand a better chance of successfully reversing the error when dealing with a large, solvent and creditworthy party than when dealing with a small, unknown and unauthorized one. Separate certificates can be issued for each counterparty in order to prevent a competitor from obtaining the user's customer list (other than himself) in a single certificate. The approved counterparty can be coded either as a common name, a distinguished name, or the hash value of either the distinguished name or the counterparty's public key. In order to claim the benefit of the transaction, the verifier must submit a certificate that matches the encoded counterparty value.

FIG. **11** shows verification by the user's sponsor of the user's transaction after receipt by a recipient. The recipient (counterparty) verifies **1110** the user's signature **1103** on the transaction **1101** and verifies **1111** the sponsor's signature **1105** on the user authorization certificate **1102**. If either of these signatures does not verify, the transaction **1101** is rejected **1112**. If the signatures verify and the transaction is accepted **1113** by the recipient, the recipient endorses the transaction **1101** by issuing his verified transaction **1114** counter-signing **1116** the text **1106** of the original user transaction **1101** and the sending user's signature **1103**, with the recipient's certificate **1115** attached. In enforcing the preapproved counterparty restriction in the sending user's authorization certificate **1102**, the sending user's sponsor verifies **1121** the sending user's signature **1103**, as included in the recipient's verified transaction **1114**, and verifies **1122** the recipient's signature **1116** thereon. If these signatures are verified, the sponsor next verifies **1123** the counterparty public key hash value by hashing the recipient's public key **1117** and checking the result against one of the authorized counterparty public key hash values **1104** as specified in the user's authorization certificate **1102** (the recipient's public key **1117** that the sponsor hashes for verification is itself verified **1124** when the sponsor verifies the recipient's certificate). If these conditions are met, the transaction is accepted **1125**.

The attribute values of delegation controls can limit the types and value ranges of authorizations that a CA may specify when issuing an attribute certificate. They can also serve to limit the scope and depth to which a user may delegate his signing authority to others. For example, a root CA might limit an organizational CA to issuing authorizations only to allow its end users to sign documents whose document types fall into a range of documents related to state tax administration. Or a CA might grant some authority to a user with the provision that it can be delegated only to another person with the rank of assistant treasurer or higher, for a time not to exceed thirty days, and without the right to further subdelegate.

Another authorization attribute, called a "confirm-to requirement" value, prevents the signature from being valid unless the verifier sends a copy of the verified transaction to a third party, typically the user's organizational sponsor or work supervisor, at a specified mail or network address, and either (a) receives an accept/reject message, or (b) a specified time elapses. This requirement is similar to a cosignature but occurs after the transaction is sent rather than before. Such after-the-fact confirmation could be acceptable in lower risk situations in which few transactions would be

rejected and in which obtaining the cosignature of the third party in advance may be unduly burdensome. As shown in FIG. **12**, the recipient first, as usual, verifies **1211** the sender's signature **1203** on the transaction **1201** and verifies **1212** the sponsor's signature **1205** on the user authorization certificate **1202**; if either of these signatures does not verify the transaction **1201** is rejected **1213**. If the signatures are verified, the recipient sends **1214** a confirmation message consisting of the original transaction **1201** (the transaction text **1202** and the sending user's signature **1203**) to the user's sponsor **1215**, as specified **1204** in the sender's authorization certificate **1202**. The recipient should receive from the sponsor **1215** the same message in return as confirmation **1216**, but signed **1205** by the sponsor. The recipient then verifies **1217** the sponsor's signature **1220** and the confirmation message **1216**, and accepts **1219** the transaction **1201**.

In order to create complex combinations of restrictions, a filter expression, which is a Boolean or logical expression involving one or more attributes, can allow construction of restrictions involving multiple attributes. The attribute assertions are linked with the usual Boolean connectives: "and", "or" and "not". For example, the sponsor might restrict a user to submitting transaction with a type equal to "purchase order" and a value less than $100,000. Assertions may involve either a single attribute value (equality, less than, greater than, etc.), multiple values of an attribute (subset, superset, etc.), or the presence or absence of an attribute in the document. Of course it will be appreciated that any or any of the described restrictions, as well as others, can be in effect at the same time for the same document or transaction. These restrictions have been discussed and illustrated separately for clarity.

The use of authorization attributes allows a recipient to verify authorization as well as authentication. In such a scenario, the sponsor certificates, anchored by the sponsoring organization's certificate, would be interpreted as authorizing "on sight" the transaction to which they are applied, assuming all specified restrictions are met.

A set of basic policies must be defined for use throughout the financial services industry and other industries in order to provide a well-defined, predictable level of service for the verification process. These policies would be agreed to on a multilateral basis by every participating firm and could stipulate that certain of the restrictions and authorizations discussed in this section would always be deemed to be in effect unless expressly provided otherwise. One of the more important elements of these industry agreements would be the definition and coding of document types. This must be done on a per-industry basis, since the rules will obviously be much different, for instance, for customs inspectors, aircraft inspectors, auditors, tax officials, etc.

Certain authorization attributes may pertain to the specific content of the document itself. This can pose problems for automated machine verification, because the verifier's computer may not always be able to determine the values of such attributes for a given document or transaction. Examples include monetary transaction limits, document types, and security or confidentiality labels. Therefore, it is desirable to provide a standard data block, preferably at the start of the document or the transaction, clearly encoding the attribute, e.g. the stated monetary transaction value, document type or security sensitivity label. This document tag will be appended by the signer's computer for the convenience of the verifier and as an aid to the verification process. However, in the event of a conflict between the tag and the actual content of the document, the language of the document would be controlling. In the case of structured transactions, such as EDI transactions, in which the document types and monetary values are already completely machine readable, document tags would not be needed.

As a possible convenience in processing simple authorizations, especially where a given user signs many similar transactions, it may often be helpful to copy the user's public key out of his basic authentication certificate and include it as another attribute in an authorization certificate. This permits the authorization certificate to serve both purposes (authentication and authorization) and allows the sender to omit the basic authentication certificate from each transaction. In addition, where a device is being relied upon to fulfill a given condition, it may likewise be advantageous to copy the user's device public key into the authorization certificate as well, further eliminating the need to send the device certificate with each transaction.

Third Party Interactions

Additional, useful features of digital signatures, beyond those that can be provided using attribute certificates, involve interaction between a signer and third parties of various types.

One such use for digital signatures is electronic notarization. As discussed above, there will be a need to cosign documents using a third party that is trusted to provide an accurate timestamp and/or location information. Simply relying upon signature originators to provide this information in an accurate fashion leaves signatures vulnerable to fraud based on, for example, pre- or post-dating of documents. An electronic "notary" would be trusted by virtue of its CA's policies to provide this information correctly. The multiple signature capabilities already assumed can be expanded to provide a framework for this service.

For notarization purposes, timestamps and location information will be included as signature attributes. Individual signature structures may either be detached and stored or, if desired, conveyed separately from the document.

Multiple signatures or joint signatures on the document itself can also be distinguished from "countersignatures," which are signatures on the signature structure in which they are found and not on the document itself. A countersignature thus provides proof of the order in which signatures were applied. Because a countersignature is itself a signature structure, it may itself contain countersignatures; this allows construction of arbitrarily long chains of countersignatures. Electronic notarization would then consist of countersigning the originator's signature and including a timestamp within the information being signed. For very high-risk applications it may also be desirable to require multiple signatures on each certificate by one or more CAs, with the signatures being performed in independent cryptographic facilities and with different private keys.

Various levels of service can be defined for electronic notaries based on the level of data verification performed prior to signing (ranging from mere existence of the document, in which case notarization may be completely automatic, to human verification of document content) and based on data retention and audit capabilities.

Another use for digital signatures is for delegation or "power of attorney" certificates. Because users are often tempted to entrust their devices or smartcards to others, for example, secretaries or co-workers, when the users go on vacation, the frequent situation, in which one user obtains another user's smartcard and PIN, exposes the smartcard to possible misuse. The system therefore facilitates the issu-

ance of power of attorney certificates that allow a delegate to associate the signature of his own smartcard with the authority of the delegating user. The power of attorney certificate would include at a minimum the name of the delegator, identification of the delegate's public key certificate and a short validity period, and would be signed by the delegator. Another possibility is for the delegate to create a new key pair exclusively for use with the delegator's signature, with the new public key included in the power of attorney certificate. This would eliminate any potential confusion between use of the delegate's private key on behalf of the delegator and on his own behalf.

The problem of handing over smart cards can be greatly reduced by providing a workable alternative that preserves the principle of individual accountability. Wide implementation of this feature will make practical the disallowance of smartcard loans, a highly desirable goal.

The use of delegation certificates discussed above implies that the user is acting as a CA. In some cases, particularly those in which the transaction crosses organizational boundaries, there may be concern that the level of controls and auditing available with the individual user's cryptographic device (e.g., a smart card) is not sufficient. In such cases, delegation certificates could be issued by a CA upon request of the delegator as normal authorization certificates. This also allows the delegation certificates to be revoked using the standard CRL mechanism. Users' certificates might then indicate a list of possible delegates, and the delegation certificate itself would contain an attribute naming the delegator.

In exercising the power of attorney, a user may indicate that he is signing for another user by including in the document or transaction a "signing-for" signature attribute, i.e., the name of the user being signed for. There must be a valid delegation certificate authorizing the signer to act for the user being signed for. Delegation is also useful in connection with a cryptographic module in a user's personal computer. Hashing and signing a document should ideally be a unitary operation in order to prevent substitution of a false hash via software hacking. However, the typical smartcard lacks the computing power to hash a very long document. One solution is to let the smartcard delegate this function to the cryptographic module using a very short-lived delegation certificate valid for only a few minutes. This certificate is signed by the user's smart card and indicates that the user of the smart card has allowed the delegation. See, for example: Gasser, M., A. Goldstein, C. Kaufman and B. Lampson, "The Digital Distributed System Security Architecture," Proceedings of the 12th National Computer Security Conference, 1989; Gasser, M. and E. McDermott, "An Architecture for Practical Delegation in a Distributed System," Proceedings of the 1990 IEEE Symposium on Security and Privacy.

A more basic problem, however, is ensuring that all possible recipients will actually employ the certificate- and attribute-verification methods described above. Although these methods allow sponsoring organizations to protect themselves, their users and those with whom they transact from liability based upon falsified transactions by allowing them to verify the identity and qualifications of those with whom they transact and the characteristics of the transactions prior to transacting, there is no guarantee that all recipients will actually so verify. If a recipient acts upon a transaction without first verifying the attributes of both the sender and the transaction, and if the sender is later found to have sent a fraudulent or unauthorized transaction, the recipient could then claim liability from the sender or its sponsor by claiming that the recipient was unaware of any requirement for authorization verification of the user's basic signature. One way to ensure that sponsors and other entities are protected from liability in such a situation is to prevent the root key, the public key of the ultimate authority, i.e., the highest-level certifying authority, which key would-be verifiers will need in order to verify any part of a transaction, from being distributed to a user (or to the user's device or smartcard) unless the user contracts with the cryptographic system and agrees to verify all parties and all transactions in accordance with the preestablished rules. In this way, the users are not technically forced to verify all parts of their transactions. However, not verifying their transactions in full would violate the contract between the users and the cryptographic system and would thereby absolve all other parties to the cryptographic system, e.g. a sponsor whose employee acted without authority, from liability. The non-verifying recipient would then bear all the risks of such an unverified transaction himself. Furthermore, because the root key of the system authority is considered a trade secret, no one who has not signed the system rules agreement may possess a copy of it, and no one could claim to have verified any part of the transaction. This art of keeping the system root key as a trade secret lends particular force and effectiveness to all the restriction and authorization methods described herein. It is believed that the possibility of incurring the potentially-large liability for valuable transactions will persuade users to employ the methods of attribute verification of this invention.

Restrictions on Certificate Distribution

Users and organizations must be able to restrict the distribution of all types of certificates for a number of reasons. First, the certificates often contain confidential business information that the user or organization prefers not be shared with others and that is nevertheless being shared with the verifier through the certificate, albeit only for the limited purpose of signature verification. Also, users' basic privacy rights may be violated if their public keys and network addresses are published. For example, they may be flooded with unsolicited business proposals and advertisements once their public keys are disseminated. Furthermore, the organization may have a general policy against giving out user identification numbers and public keys, because they may be used as starting points for various types of security attacks.

This functionality may be implemented in the user's attribute certificate. If the "distribution-restriction" attribute is TRUE, the user/issuer grants permission to use the certificate and its associated public key certificate only for signature verification; distribution or further publication is prohibited. Other ways to specify this restriction might include placing the attribute in the organization's certificate, publishing the restriction as part of the industry-specific policy, or (in a true X.500 implementation) using the X.500 access control list mechanism to restrict casual access to the certificate. Although some existing general legal basis for enforcing this restriction might be found under copyright law, i.e., if the certificate is declared as an unpublished work for which a license is granted only to the named verifier, a firmer legal basis will still be desirable.

Smartcard Requirements

There are some additional requirements on smartcards when used with commercial digital signature systems.

The first requirement is private key confinement and self-certification. That is, the user's private signature key

must never be allowed to leave the smart card. Only in this way can it be assured that theft of the key cannot be accomplished through purely electronic means without leaving any evidence. This principle of private key confinement is vital to the concept of non-repudiation.

Thus, as illustrated in FIG. **13**, when providing a public key **1303** to be certified, the card **1301** must attest that the card **1301** is tamperproof and possesses a key confining design. Proof can be provided via a "device certificate" **1302** stating that the card originates from the specific manufacturer or product line. The public key **1308** of the device **1301** must then be certified by the manufacturer or by a CA designated by the manufacturer. One likely approach to creating this device certificate would be to generate the device key pair during fabrication of the smartcard so that the corresponding device certificate **1302** could also be included on the card. The device certificate **1302** certifies the properties **1304** of the card, and the card generates a key pair **1303,1309** which is to be used by the user of the card and which the user can have certified as his own by any appropriate desired CA. Then, when submitting a newly generated public key **1303** for certification, the device private signature key **1305** would be used to countersign **1306** the certificate request data **1307**, which is already signed by the newly-generated user private key **1309**.

Also, in a case in which the government requires that all decryption keys be escrowed, the card should be able to certify that it is incapable of decryption. This "signature only" certification can be implemented through the same mechanisms described above, thus allowing the user's signature key to remain exempt from escrow requirements. Because it is doubtful whether an escrowed key retains any value for non-repudiation services, this certification is vital in order to prevent the signature key's disclosure through possible mishandling during an escrow process.

Smartcards should also be required to guard against unauthorized use of personal identification numbers (PINs). Normally, a smartcard is protected against unauthorized use by a PIN, the equivalent of a password. Typically, a PIN is changeable only by the user and must be a specified length, but typically nothing prevents the user from setting the PIN to a trivial number, e.g. all 1's or 121212. Smartcard vendors should be requested to implement PIN-change routines that insure non-trivial PINs without repeating digits or obvious patterns. Making the PIN relatively long (at least 6 digits) and non-trivial reduces the chance that the card can be operated by someone finding or stealing it. Support for a 6-digit PIN requirement can be found in ANSI, "X9.26: Financial Institution Sign-On Authentication for Wholesale Financial Transactions", 1990, which is hereby incorporated by reference and which sets forth the "one-in-a-million" standard that states that a log-in mechanism may be considered secure if, among other things, an attacker has no more than a one-in-a-million chance of guessing the correct password and if the system takes evasive action to prevent repeated guessing. Furthermore, smartcards should be required to take "evasive action", e.g., shutting down for a period of time or even erasing private keys, if too many incorrect PINs are entered by an attempted user.

It could also be made a requirement that smartcard manufacturers use biometrics as more secure methods of identification. Extensive work is currently being done in the areas of voiceprint and fingerprint identification, as a supplement to PINs. However, while the rates of false positive and negative still must be reduced, the main problem lies in securing the biometric input device and its data channel so that they are immune to capture and replay of the biometric data. This is not a problem when the biometric device is embedded in a concrete wall, for example in an ATM or door access system, but it remains a serious problem in typical commercial office settings. Ideally, the card and biometric input device will each be tamperproof cryptographic modules that can certify themselves and establish secure channels with each other.

Smartcards should also be able to maintain an "audit trail," or an internal log of recent actions, containing at a minimum, a timestamp, transaction amount, type code and message digest. This information can be compressed into 40 or so bytes so that a 400-record circular log would consume around 16K bytes. This log would be uploaded and checked only on receipt of a signed request from the card issuer over a secure channel. Also, the card would not delete the old log until it received a signed confirmation from the issuer stating that the uploaded log had been received intact. This control mechanism will deter forgery, reduce the damage that can be caused by a forger, and allow unauthorized or questioned transactions to be investigated more quickly and easily. Since most or all transactions occur off-line from the issuer, the card is the best witness of its own actions.

Thus, a method for securely using digital signatures in a commercial cryptographic system is provided. One skilled in the art will appreciate that the present invention can be practiced by other that the described embodiments, which are presented for purposes of illustration and not limitation, and the present invention is limited only by the claims that follow.

I claim:

1. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and attribute data, and a digital user signature based on said digital message and on a private key of a user;

receiving a digital identifying certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user;

receiving a digital authorizing certificate, separate from said identifying certificate and issued by a sponsor of said user, and authorizing transactions by said user, said authorizing certificate containing rules specifying conditions under which said digital transaction is valid, said rules to be applied to said attribute data;

verifying said transaction based on information in said identifying certificate and in said authorizing certificate, said step of verifying including applying said rules to said attribute data in order to verify that said transaction is valid; and

accepting said transaction based on said outcome of said verifying,

wherein said attribute data includes a timestamp indicating when said transaction was formed and wherein said rules specify allowed times at which transactions can be formed and wherein said step of verifying includes a step of determining whether said timestamp indicates one of said specified allowed times, and

wherein said allowed times are certain days of the week, and wherein said step of verifying includes a step of determining whether said timestamp indicates that said transaction was formed on one of said certain days of the week.

2. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and attribute data, and a digital user signature based on said digital message and on a private key of a user;

receiving a digital identifying certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user;

receiving a digital authorizing certificate, separate from said identifying certificate and issued by a sponsor of said user, and authorizing transactions by said user, said authorizing certificate containing rules specifying conditions under which said digital transaction is valid, said rules to be applied to said attribute data;

verifying said transaction based on information in said identifying certificate and in said authorizing certificate, said step of verifying including applying said rules to said attribute data in order to verify that said transaction is valid; and

accepting said transaction based on said outcome of said verifying,

wherein said attribute data includes a timestamp indicating when said transaction was formed and wherein said rules specify allowed times at which transactions can be formed and wherein said step of verifying includes a step of determining whether said timestamp indicates one of said specified allowed times, and

wherein said allowed times are certain times of day, and wherein said step of verifying includes a step of determining whether said timestamp indicates that said transaction was formed at said certain times of day.

3. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and attribute data, and a digital user signature based on said digital message and on a private key of a user;

receiving a digital identifying certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user;

receiving a digital authorizing certificate, separate from said identifying certificate and issued by a sponsor of said user, and authorizing transactions by said user, said authorizing certificate containing rules specifying conditions under which said digital transaction is valid, said rules to be applied to said attribute data;

verifying said transaction based on information in said identifying certificate and in said authorizing certificate, said step of verifying including applying said rules to said attribute data in order to verify that said transaction is valid; and

accepting said transaction based on said outcome of said verifying,

wherein said attribute data includes a timestamp indicating when said transaction was formed and wherein said rules specify a time period after said transaction was formed within which said user signature is valid, and wherein said transaction is invalid if said signature is not verified within said specified time period, and wherein said step of verifying said transaction comprises the steps of:

verifying said signature and

determining whether said verifying of said signature took place within said specified time period.

4. A method A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and attribute data, and a digital user signature based on said digital message and on a private key of a user;

receiving a digital identifying certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user;

receiving a digital authorizing certificate, separate from said identifying certificate and issued by a sponsor of said user, and authorizing transactions by said user, said authorizing certificate containing rules specifying conditions under which said digital transaction is valid, said rules to be applied to said attribute data;

verifying said transaction based on information in said identifying certificate and in said authorizing certificate, said step of verifying including applying said rules to said attribute data in order to verify that said transaction is valid; and

accepting said transaction based on said outcome of said verifying,

wherein said attribute data includes a role said user is exercising by performing said transaction and wherein said rules specify roles which said user may exercise, and wherein said step of verifying includes a step of determining whether said role is one of said specified roles.

5. A method A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and attribute data, and a digital user signature based on said digital message and on a private key of a user;

receiving a digital identifying certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user;

receiving a digital authorizing certificate, separate from said identifying certificate and issued by a sponsor of said user, and authorizing transactions by said user, said authorizing certificate containing rules specifying conditions under which said digital transaction is valid, said rules to be applied to said attribute data, wherein said rules specify a list of at least one recipient of said transaction considered acceptable by said sponsor;

verifying said transaction based on information in said identifying certificate and in said authorizing certificate, said step of verifying including applying said rules to said attribute data in order to verify that said transaction is valid, said step of verifying by a recipient including the step of determining whether said recipient is in said list; and

accepting said transaction based on said outcome of said verifying,

said method further comprising the steps of, after said step of verifying:

forming a digital recipient signature based on said user transaction and on a private key of said recipient;

combining said digital recipient signature and said user transaction to form a verified transaction;

providing to said sponsor said verified transaction.

6. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and attribute data, and a digital user signature based on said digital message and on a private key of a user;

receiving a digital identifying certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user;

receiving a digital authorizing certificate, separate from said identifying certificate and issued by a sponsor of said user, and authorizing transactions by said user, said authorizing certificate containing rules specifying conditions under which said digital transaction is valid, said rules specifying that said sponsor must be notified of and must approve said transaction, said rules to be applied to said attribute data;

verifying said transaction based on information in said identifying certificate and in said authorizing certificate, said step of verifying including applying said rules to said attribute data in order to verify that said transaction is valid;

accepting said transaction based on said outcome of said verifying;

notifying said sponsor of said transaction; and

awaiting a reply from said sponsor.

7. A method as in claim **6**, wherein said step of notifying includes the steps of:

sending said user transaction to said sponsor, and wherein said sponsor indicates approval of said transaction by returning to said recipient a confirmation transaction formed by said sponsor digitally signing said sent user transaction; and wherein said step of verifying further comprises the steps of:

receiving from said sponsor a reply and determining whether said reply is a confirmation transaction signed by said sponsor.

8. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

forming a digital message by a user;

forming a digital user signature based on said digital message and a private key of said user;

combining said digital message and said digital user signature to form a digital user transaction, said digital transaction containing attribute data including a timestamp indicating when said transaction was formed;

combining with said digital user transaction a digital identifying certificate issued by a certifying authority, said identifying certificate having a plurality of digital fields, at least one of said fields identifying said user; and

combining with said digital transaction a digital authorizing certificate, separate from said identifying certificate and issued by a sponsor of said user, for authorizing transactions by said user, wherein said digital authorizing certificate contains rules specifying conditions under which said digital transaction is valid, said rules to be applied to said attribute data in order to determine whether said transaction is valid,

wherein said rules specify allowed times at which transactions can be formed, and wherein said transaction is invalid if said timestamp does not indicate one of said specified allowed times, and

wherein said allowed times are certain days of the week, and wherein said transaction is invalid if said timestamp indicates that said transaction was not formed on one of said certain days of the week.

9. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

forming a digital message by a user;

forming a digital user signature based on said digital message and a private key of said user;

combining said digital message and said digital user signature to form a digital user transaction, said digital transaction containing attribute data including a timestamp indicating when said transaction was formed;

combining with said digital user transaction a digital identifying certificate issued by a certifying authority, said identifying certificate having a plurality of digital fields, at least one of said fields identifying said user; and

combining with said digital transaction a digital authorizing certificate, separate from said identifying certificate and issued by a sponsor of said user, for authorizing transactions by said user, wherein said digital authorizing certificate contains rules specifying conditions under which said digital transaction is valid, said rules to be applied to said attribute data in order to determine whether said transaction is valid,

wherein said rules specify allowed times at which transactions can be formed, and wherein said transaction is invalid if said timestamp does not indicate one of said specified allowed times, and

wherein said allowed times are certain times of day and wherein said transaction is invalid if said timestamp indicates that said transaction was not formed at said certain times of day.

10. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

forming a digital message by a user;

forming a digital user signature based on said digital message and a private key of said user;

combining said digital message and said digital user signature to form a digital user transaction, said digital transaction containing attribute data;

combining with said digital user transaction a digital identifying certificate issued by a certifying authority, said identifying certificate having a plurality of digital fields, at least one of said fields identifying said user; and

combining with said digital transaction a digital authorizing certificate, separate from said identifying certificate and issued by a sponsor of said user, for authorizing transactions by said user, wherein said digital authorizing certificate contains rules specifying conditions under which said digital transaction is valid, said rules to be applied to said attribute data in order to determine whether said transaction is valid,

wherein said attribute data includes a role said user is exercising by performing said transaction and wherein said rules specify roles which said user may exercise and wherein said transaction is invalid if said role is not one of said specified roles.

11. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

forming a digital message by a user;

forming a digital user signature based on said digital message and a private key of said user;

combining said digital message and said digital user signature to form a digital user transaction, said digital transaction containing attribute data;

combining with said digital user transaction a digital identifying certificate issued by a certifying authority, said identifying certificate having a plurality of digital fields, at least one of said fields identifying said user; and

combining with said digital transaction a digital authorizing certificate, separate from said identifying certifi-

cate and issued by a sponsor of said user, for authorizing transactions by said user, wherein said digital authorizing certificate contains rules specifying conditions under which said digital transaction is valid, said rules to be applied to said attribute data in order to determine whether said transaction is valid, wherein said rules specify that said sponsor must be notified of and must approve said transaction and wherein said transaction is invalid unless said sponsor is notified of said transaction.

12. A method of enforcing a policy in a cryptographic communication system comprising:

forming a digital message by a user;

forming a digital user signature based on said digital message and a private key of said user;

combining said digital message and said digital user signature to form a digital user transaction;

combining with said digital user transaction a digital identifying certificate issued by a certifying authority, said identifying certificate having a plurality of digital fields, at least one of said fields identifying said user; and

combining with said digital transaction a digital authorizing certificate, separate from said identifying certificate and issued by a sponsor of said user, for authorizing transactions by said user,

wherein said digital authorizing certificate contains rules identifying conditions under which said digital transaction is valid, said rules to be applied to said digital message in order to determine whether said transaction is valid.

13. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

forming a digital message by a user;

forming a digital user signature based on said digital message and a private key of said user;

combining said digital message and said digital user signature to form a digital user transaction, wherein said transaction contains attribute data including a document type of said message;

combining with said digital user transaction a digital certificate issued by a certifying authority, said certificate having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, said rule to be applied to said attribute data in order to determine whether said transaction is valid, and wherein said rule specifies allowed document types for messages, and wherein said transaction is invalid if said document type is not one of said specified allowed document types.

14. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and a digital user signature based on said digital message and on a private key of a user, wherein said transaction contains attribute data including a document type of said message;

receiving a digital certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, said rule to be applied to said attribute data in order to determine whether said transaction is valid, and wherein said rule specifies allowed document types for messages;

verifying said transaction based on information in said certificate, including determining whether said document type is one of said specified allowed document types; and

accepting said transaction based on said outcome of said verifying.

15. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

forming a digital message by a user;

forming a digital user signature based on said digital message and a private key of said user;

combining said digital message and said digital user signature to form a digital user transaction, wherein said transaction contains attribute data including a location at which said transaction was formed;

combining with said digital user transaction a digital certificate issued by a certifying authority, said certificate having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, said rule to be applied to said attribute data in order to determine whether said transaction is valid, and wherein said rule specifies allowed locations at which transactions may be formed, and wherein said transaction is invalid if said location is not one of said specified allowed locations.

16. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and a digital user signature based on said digital message and on a private key of a user, wherein said transaction contains attribute data including a location at which said transaction was formed;

receiving a digital certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, and wherein said rule specifies allowed locations at which transactions may be formed;

verifying said transaction based on information in said certificate, including determining whether said location is one of said specified allowed locations; and

accepting said transaction based on said outcome of said verifying.

17. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

forming a digital message by a user;

forming a digital user signature based on said digital message and a private key of said user;

combining said digital message and said digital user signature to form a digital user transaction, wherein said transaction contains attribute data including a timestamp indicating when said transaction was formed;

combining with said digital user transaction a digital certificate issued by a certifying authority, said certificate having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition

under which said digital transaction is valid according to said certifying authority, said rule to be applied to said attribute data in order to determine whether said transaction is valid, and wherein said rule specifies allowed times at which transactions may be formed, and wherein said transaction is invalid if said timestamp does not indicate one of said specified allowed times.

18. A method as in claim **17**, wherein said allowed times are certain days of the week, and wherein said transaction is invalid if said timestamp indicates that said transaction was not formed on one of said certain days of the week.

19. A method as in claim **17**, wherein said allowed times are certain times of day and wherein said transaction is invalid if said timestamp indicates that said transaction was not formed at said certain times of day.

20. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and a digital user signature based on said digital message and on a private key of a user, wherein said transaction contains attribute data including a timestamp indicating when said transaction was formed;

receiving a digital certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, and wherein said rule specifies allowed times at which transactions may be formed;

verifying said transaction based on information in said certificate, including determining whether said timestamp indicates one of said specified allowed times; and

accepting said transaction based on said outcome of said verifying.

21. A method as in claim **20**, wherein said allowed times are certain days of the week, and wherein said step of verifying includes a step of determining whether said timestamp indicates that said transaction was formed on one of said certain days of the week.

22. A method as in claim **20**, wherein said allowed times are certain times of day, and wherein said step of verifying includes a step of determining whether said timestamp indicates that said transaction was formed at said certain times of day.

23. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

forming a digital message by a user;

forming a digital user signature based on said digital message and a private key of said user;

combining said digital message and said digital user signature to form a digital user transaction, wherein said transaction contains attribute data including a timestamp indicating when said transaction was formed;

combining with said digital user transaction a digital certificate issued by a certifying authority, said certificate having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, said rule to be applied to said attribute data in order to determine whether said transaction is valid, and wherein said rule specifies a time period within which said signature is valid, and wherein said transaction is invalid if said signature is not verified within said specified time period.

24. A method as in claim **23**, wherein said specified time period specifies a maximum allowable age of said user signature, and wherein said transaction is invalid if said user signature is not verified before said user signature reaches said maximum allowable age.

25. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and a digital user signature based on said digital message and on a private key of a user, wherein said transaction contains attribute data including a timestamp indicating when said transaction was formed;

receiving a digital certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, and wherein said rule specifies a time period within which said signature is valid;

verifying said transaction based on information in said certificate; and

accepting said transaction based on said outcome of said verifying,

wherein said step of verifying includes the steps of:

verifying said signature; and

determining whether said verifying of said signature took place within said specified time period.

26. A method as in claim **25**, wherein said specified time period specifies a maximum allowable age of said user signature, and wherein said transaction is invalid if said user signature is not verified before said user signature reaches said maximum allowable age.

27. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

forming a digital message by a user;

forming a digital user signature based on said digital message and a private key of said user;

combining said digital message and said digital user signature to form a digital user transaction, wherein said transaction contains attribute data including a role said user is exercising by performing said transaction;

combining with said digital user transaction a digital certificate issued by a certifying authority, said certificate having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, said rule to be applied to said attribute data in order to determine whether said transaction is valid, and wherein said rule specifies allowed roles which said user may exercise, and wherein said transaction is invalid if said role is not one of said specified allowed roles.

28. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and a digital user signature based on said digital message and on a private key of a user, wherein said transaction contains attribute data including a role said user is exercising by performing said transaction;

receiving a digital certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, and wherein said rule specifies allowed roles which said user may exercise;

verifying said transaction based on information in said certificate, including determining whether said role is one of said specified allowed roles; and

accepting said transaction based on said outcome of said verifying.

29. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

forming a digital message by a user;

forming a digital user signature based on said digital message and a private key of said user;

combining said digital message and said digital user signature to form a digital user transaction;

combining with said digital user transaction a digital certificate issued by a certifying authority, said certificate having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, said rule to be applied to said attribute data in order to determine whether said transaction is valid, and wherein said rule specifies that a specified entity must be notified of said transaction and wherein said transaction is invalid unless said specified entity is notified of and approves said transaction.

30. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and a digital user signature based on said digital message and on a private key of a user;

receiving a digital certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, and wherein said rule specifies that a specified entity must be notified of said transaction;

verifying said transaction based on information in said certificate;

accepting said transaction based on said outcome of said verifying;

notifying said specified entity of said transaction; and

awaiting a reply from said specified entity.

31. A method as in claim **30**, wherein said step of notifying includes the step of:

sending said user transaction to said specified entity, and wherein said specified entity indicates approval of said transaction by returning to said recipient a confirmation transaction formed by said specified entity digitally signing said sent user transaction; and wherein said step of verifying further comprises the steps of:

receiving from said specified entity a reply and determining whether said reply is a confirmation transaction signed by said specified entity.

32. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

forming a digital message by a user;

forming a digital user signature based on said digital message and a private key of said user;

combining said digital message and said digital user signature to form a digital user transaction;

combining with said digital user transaction a digital certificate issued by a certifying authority, said certificate having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, said rule to be applied to said attribute data in order to determine whether said transaction is valid, and wherein said rule specifies a list of at least one recipient of said transaction considered acceptable by said certifying authority and wherein said transaction is invalid if it is acted on by a recipient not in said list.

33. A method as in claim **31**, wherein said transaction is invalid unless said certifying authority determines that said recipient is in said list.

34. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and a digital user signature based on said digital message and on a private key of a user;

receiving a digital certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, and wherein said rule specifies a list of at least one recipient of said transaction considered acceptable by said certifying authority;

verifying said transaction based on information in said certificate, including determining whether a recipient is in said list, and

accepting said transaction based on said outcome of said verifying.

35. A method as in claim **34**, wherein said transaction is invalid unless said certifying authority determines that said recipient is in said list, said method further comprising the steps of, after said step of verifying:

forming a digital recipient signature based on said user transaction and on a private key of said recipient;

combining said digital recipient signature and said user transaction to form a verified transaction;

providing to said certifying authority said verified transaction.

36. A method as in claim **11**, wherein said transaction is invalid unless, upon notification of said transaction, said sponsor approves said transaction.

37. A method of enforcing a policy in a cryptographic communication system comprising the steps of:

receiving a digital user transaction including a digital message and a digital user signature based on said digital message and on a private key of a user, wherein said transaction contains attribute data including a timestamp indicating when said transaction was formed;

receiving a digital certificate issued by a certifying authority and having a plurality of digital fields, at least one of said fields identifying said user and at least one other of said fields containing a rule specifying a condition under which said digital transaction is valid according to said certifying authority, and wherein said rule specifies a time period after said transaction was formed within which said user signature is valid, and wherein said transaction is invalid if said signature is not verified within said specified time period,

verifying said transaction based on information in said certificate; and

accepting said transaction based on said outcome of said verifying,

wherein said step of verifying includes the steps of:

verifying said signature; and

determining whether said verifying of said signature took place after said specified time period.

* * * * *